

CP/M Handbuch

Rodnay Zaks

Rodnay Zaks

CP/M HANDBUCH



BERKELEY · PARIS · DÜSSELDORF

Anmerkungen:

CP/M ist ein geschütztes Warenzeichen von Digital Research, USA.
MP/M ist ein Warenzeichen von Digital Research, USA.
NAD, QSORT sind geschützte Warenzeichen von Structured Systems Group Inc., USA.
WORDMASTER, WORDSTAR sind geschützte Warenzeichen von Micropro International Corporation, USA.
MDS ist ein geschütztes Warenzeichen von Intel Corporation, USA.
Z8000, Z80 sind geschützte Warenzeichen von Zilog Inc., USA.
TRS80 ist ein geschütztes Warenzeichen von Tandy Corporation, USA.
APPLE ist ein geschütztes Warenzeichen von Apple Inc., USA.
PET, CBM sind geschützte Warenzeichen von Commodore Inc., USA.

Originalausgabe in Englisch.
Titel der englischen Ausgabe: The CP/M handbook with MP/M

Deutsche Übersetzung: Dr. Klaus-Jürgen Schmidt
Überarbeitung: Günter Mußtopf und Winfried Wolf

Umschlagentwurf: Daniel Boucherie
Zeichnungen: Daniel Le Noury
Technische Illustrationen: J. Trujillo Smith
Satz: tgr – typo-grafik-repro gmbh, Remscheid
Gesamtherstellung: Druckerei Hub. Hoch, Düsseldorf

ISBN 3-88745-053-1
1. Auflage 1981
2. Auflage 1981
3. Auflage 1982
4. Auflage 1983
5. Auflage 1984 (2., überarbeitete und erweiterte Ausgabe)
6. Auflage 1985
7. Auflage 1986

Printed in Germany

Inhalt

Vorwort

Mit diesem Buch wird der Einsatz der CP/M-Familie und deren weitreichende Anwendungsmöglichkeiten vermittelt. Hierbei werden zwar keine Vorkenntnisse über Computer vorausgesetzt, jedoch sollte ein mit CP/M lauffähiges Computersystem zur Verfügung stehen.

CP/M kann heute als Standardbetriebssystem für Mikrocomputer betrachtet werden. Die meisten Anwender von Computersystemen auf der Basis von Mikroprozessoren werden früher oder später mit CP/M konfrontiert. Abhängig vom jeweiligen Anwenderprogramm werden dabei einige oder mehrere Funktionen vom CP/M genutzt. So benötigt z.B. ein Datentypist zur Eingabe von Buchhaltungsdaten nur Kenntnisse über den Start des Datenerfassungsprogramms, über dessen Fehlermeldungen und über die Korrektur dieser Fehler. Von einem erfahrenen Programmierer werden darüber hinaus noch Kenntnisse über die Entwicklung und Installation von Programmsystemen erwartet. Dieses Buch ist deshalb so strukturiert, daß ein breites Spektrum von Nutzungsmöglichkeiten abgedeckt wird.

Kapitel 1 führt in CP/M ein. In ihm werden mit einfachen Beispielen alle Grundoperationen, beginnend vom Einschalten des Computers bis zum Duplizieren von Disketten, erläutert. Es werden hierzu zunächst einige Grundkenntnisse über Hardware- und Softwarekomponenten vermittelt. Weiterhin werden das Einschalten und Starten von Rechnersystemen, das Starten von Programmen, das Generieren von Dateien, das Kopieren von Dateien und Disketten sowie weitere wichtige Befehle zur Dateibearbeitung besprochen. Auf diese Weise werden alle wesentlichen Operationen behandelt, mit denen ein Anwender bei der Abwicklung seiner Programme konfrontiert wird und die für ein störungsfreies Arbeiten mit CP/M notwendig sind. In einer erstaunlich kurzen Zeit ist ein Benutzer damit so weit vorbereitet, daß er seinen Computer mit CP/M wirkungsvoll einsetzen kann.

Kapitel 2 bietet die Grundlage für die Anwendung der Betriebssysteme der CP/M-Familie. In ihm werden spezifische Informationen über die Struktur der CP/M-Befehle gegeben und alle z.Z. nutzbaren CP/M-Befehle in komprimierter Form behandelt. Es werden hier zwar nicht alle Details erfaßt, dennoch ergibt sich ein Überblick über alle von CP/M und MP/M unterstützten Aufgaben.

Für einen erfahrenen CP/M-Anwender sind vollständige Kenntnisse über ein ihm verfügbares, leistungsfähiges Dateiverwaltungsprogramm uner-

läßlich. In *Kapitel 3* werden hierzu alle Funktionen des „Peripheral Interchange Program" (PIP) beschrieben wie z.B. das Mischen, Kopieren und Auflisten von Dateiinhalten und Dateigruppen.

Kapitel 4 erläutert mit Hilfe eines fortlaufenden Textbeispiels die Möglichkeiten des CP/M-Editierprogramms ED. Trotz der internen Komplexität dieses Programms ist es relativ schnell anwendbar und effektiv nutzbar.

In den bis hierhin behandelten Kapiteln sind alle Möglichkeiten von CP/M vorgestellt worden und ohne weitere Informationen anwendbar. Im *Kapitel 5* wird danach die interne Struktur mit den internen Operationen von CP/M erklärt. Die Kenntnisse der Struktur sind zwar für die Anwendung von CP/M überflüssig, jedoch bei gewünschter Änderung der Systemkonfiguration notwendig.

Im *Kapitel 6* sind nochmals alle besprochenen Befehle und Programme zusammengefaßt und in einheitlicher Form erklärt.

Wichtige Informationen und praktische Hinweise für die Anwendung des gesamten Computersystems werden in *Kapitel 7* gegeben. Hierbei stehen konkrete Anweisungen für die Behandlung des Rechners, der Peripherieeinheiten, der Disketten usw. und die Raumausstattung im Vordergrund. Dieses Kapitel ist deshalb für jeden Anwender für seine tägliche praktische Arbeit am System von besonderer Bedeutung.

Kapitel 8 gibt einen kurzen geschichtlichen Überblick über die Entwicklung von CP/M und dessen Zukunft.

Im *Anhang* sind häufig benötigte Informationen enthalten. Sie geben dem Benutzer in tabellarischer Darstellung einen Überblick über wichtige Binärcodes, Fehlermeldungen, Symbole und Befehle des CP/M, PIP und ED. Weiterhin werden Hinweise auf interessante Dienstprogramme gegeben.

CP/M wurde zur problemlosen Nutzung von Mikrocomputern entwickelt. Das Buch „CP/M-Handbuch" soll diese Nutzung so schnell wie möglich und so umfangreich wie notwendig in kürzester Zeit vermitteln helfen. Es bezieht sich auf die verbreiteten CP/M-Versionen CP/M 2.2 (CP/M-80), CP/M-86, CP/M-Plus, Concurrent CP/M-86 (CCP/M-86), MP/MII (MP/M-80) und MP/M-86, ist jedoch mit gleicher Effektivität auf alle anderen CP/M-verträglichen Betriebssysteme wie z.B. CDOS von Cromemco, EOS von Sharp, MOS von Mostek oder TurboDOS anwendbar.

Vorwort zum erweiterten CP/M-Handbuch

Seit dem ersten Erscheinen des CP/M-Handbuchs im Jahre 1980 hat sich das Betriebssystem CP/M zu einer außerordentlich erfolgreichen Familie von Betriebssystemen entwickelt:

- CP/M-Plus (CP/M 3.0) für 8-Bit-Systeme auf der Basis von 8080/8085 und Z80 mit wesentlich verbesserter Leistung verglichen mit CP/M 2.2, und
- Concurrent CP/M-86 für 16-Bit-Systeme auf der Basis von 8086/8088. Im Unterschied zu CP/M 2.2 (bzw. CP/M-86 und CP/M-Plus) können mehrere Aufgaben (Tasks) gleichzeitig bearbeitet werden.

Die neuen Mitglieder der CP/M-Familie wurden in der Zwischenzeit in Verbindung mit Personal Computern auf den Markt gebracht. Eine Ergänzung und Erweiterung des CP/M-Handbuchs war deshalb dringend geboten. Die neuen und aus der Sicht des Anwenders wichtigen Eigenschaften und Funktionen wurden in den betreffenden Kapiteln ergänzt. Dagegen wurde CP/M 1.4 gestrichen, da es heute kaum noch eingesetzt wird.

Der Leistungsumfang insbesondere der neuen „CP/M Betriebssysteme" wie CP/M-Plus und CCP/M (zur Zeit Concurrent CP/M-86) ist verglichen mit CP/M 2.2 entscheidend gesteigert worden. Dies bedeutet gleichzeitig, daß in einem solchen Handbuch keine Vollständigkeit erreicht werden kann. Es werden jedoch alle für die Benutzung bzw. Bedienung erforderlichen Funktionen und Handgriffe behandelt. Nach dem Studium dieses Handbuchs sollte der Programmierer sein Wissen mit Büchern wie z.B. *A. Miller, Mastering CP/M, SYBEX INC., Berkeley, USA* erweitern.

Außer den für die Betriebssysteme selbst erforderlichen Ergänzungen wurden folgende Teile des CP/M-Handbuchs überarbeitet bzw. neu hinzugefügt:

- *Kapitel 7:* Praktische Hinweise. Eine Reihe interessanter Dienstprogramme wird von verschiedenen Herstellern angeboten, die die Arbeit mit CP/M erleichtern. Der Leistungsumfang, das Einsatzgebiet und die Bedienung einiger wichtiger Dienstprogramme werden kurz beschrieben.

- *Anhang A:* CP/M-Fehlermeldungen. Dieser Anhang wurde ebenfalls ergänzt und durch die Fehlermeldungen der von Digital Research gelieferten Dienstprogramme erweitert.
- *Anhang P:* Begriffe und Abkürzungen. Die „Unsitte" des Gebrauchs englischer Fachbegriffe und Abkürzungen läßt sich leider nicht ganz vermeiden. Der „CP/M-Neuling" findet hier die im CP/M-Handbuch häufig verwendeten Begriffe und Abkürzungen mit kurzen Erklärungen.
- *Anhang Q:* Wichtige Adressen. Insbesondere in Kapitel 7 werden mehrere Produkte beschrieben. Im Anhang Q sind einige in diesem Zusammenhang interessante Adressen angegeben.

Die Überarbeitung der erweiterten deutschen Ausgabe wurde von Herrn Günter Mußtopf, Ahrensburg, und Herrn Winfried Wolf, Hamburg, durchgeführt.

Hamburg, Februar 1984

Kapitel 1
Einführung in CP/M und MP/M

EINLEITUNG

In diesem Kapitel werden alle wesentlichen, mit CP/M an einem Mikrocomputer durchführbaren Systemaufgaben vorgestellt. Hierzu sind keine besonderen Kenntnisse über Computersysteme erforderlich. Zunächst werden einige Grundlagen über Computer und deren Grundkomponenten erläutert. Danach wird besprochen, wie ein Computersystem eingeschaltet, die System-Diskette eingelegt und CP/M gestartet wird. Im nächsten wichtigen Abschnitt werden Dateien (Files) vorgestellt, generiert, Dateibezeichnungen zugeordnet, einzelne Dateien und ganze Disketteninhalte kopiert. Darüber hinaus wird erläutert, wie die Tastatur des Bildschirmterminals sowie der Drucker zu bedienen sind und hierüber Inhalte einzelner Dateien gelistet werden können. Nach Abschluß des Kapitels sind dann die wichtigsten CP/M-Befehle bekannt und können bereits sinnvoll eingesetzt werden.

GRUNDLAGEN

Abb. 1.1 zeigt ein heute in Deutschland eingesetztes, typisches Mikrocomputersystem. Dieses System setzt sich aus dem im Computer enthaltenen Bildschirm, der Tastatur, den integrierten Diskettenlaufwerken und dem Drucker zusammen. Der Computer wird über die Tastatur bedient (beschickt), auf dessen Bildschirm die entsprechenden Meldungen zu sehen sind. Mit Hilfe des Druckers kann dann jeder beliebige Text ausgedruckt werden. Die vom Computer auszuführenden Programme werden auf einer Diskette gespeichert, die in eines der Laufwerke eingeschoben wurde. In diesem Kapitel wird Schritt für Schritt erklärt, wie alle für das Computersystem notwendigen Operationen ausgeführt werden.

DAS COMPUTERSYSTEM

Ein Computersystem besteht aus *Hardware* und *Software*. Die Hardware umfaßt alle physikalischen Komponenten des Systems (ICs, Platinen, Schrauben, Muttern, Drahtverbindungen usw.). Die Software umfaßt die Programme und Dateien.

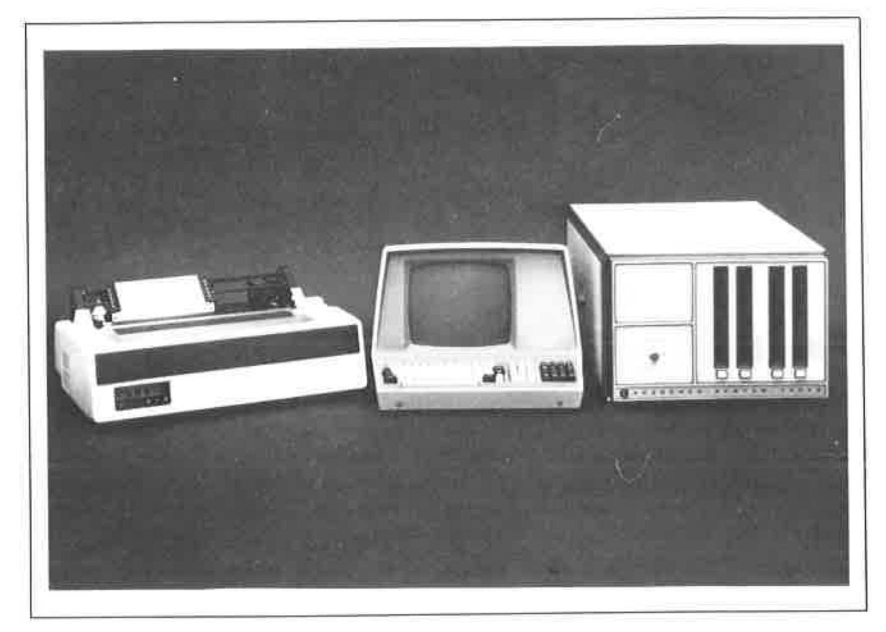


Abb. 1.1: Typisches Mikrocomputersystem

Die Hardware-Komponenten

Abb. 1.1 zeigt die Hardwarekomponenten eines typischen Personal Computers (Computer mit integriertem Bildschirm und Diskettenlaufwerken, Tastatur und Drucker). Zusätzliche Hardwarekomponenten wie Magnetband-Kassetten oder andere Geräte (Lochstreifenleser/-stanzer, Lichtgriffel, Joystick usw.) können noch hinzugeschaltet werden.

Der Computer

Während anfangs die Mikrocomputer aus einem separaten Gehäuse mit integrierten Diskettenlaufwerken und einem Bildschirm-Terminal mit Tastatur bestanden, werden heute vielfach Computer und Diskettenlaufwerke in die Bildschirmeinheit eingebaut (z.B. NCR Decision Mate V). Insbesondere bei den 16-Bit-Systemen wie dem IBM PC sind in einem Gehäuse Computer, Laufwerke und Bildschirmsteuerung enthalten. Getrennt davon sind der Bildschirm-Monitor und die Tastatur (Abb. 1.2). In anderen Systemen ist der Computer in die Tastatur integriert. Getrennt davon sind der Bildschirm-Monitor und Laufwerke erforderlich (z.B. Triumph-Adler PC; Abb. 1.3).

Die Hauptaufgabe eines Computers liegt darin, Informationen zu verarbeiten. Die hierzu notwendigen Operationen werden von Programmen

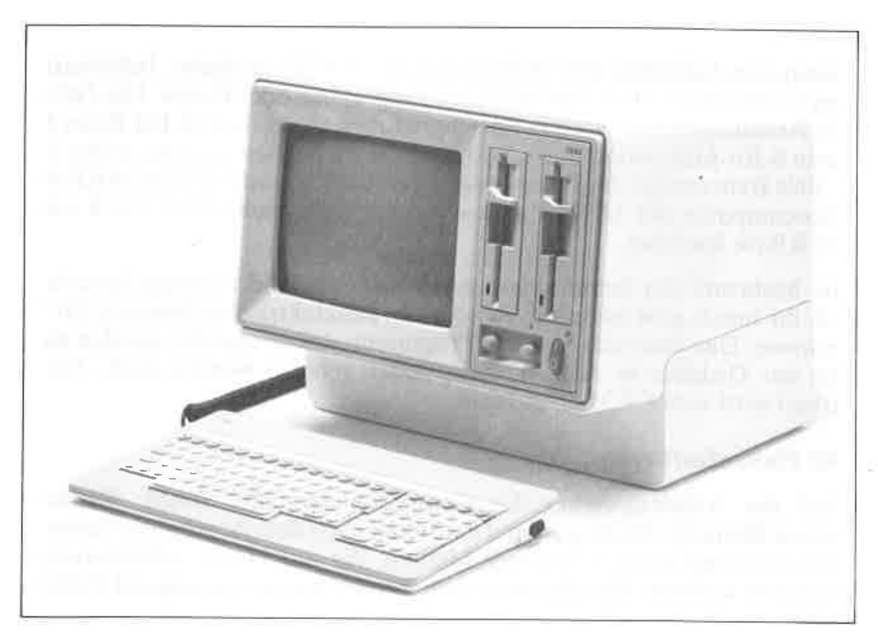


Abb. 1.2: NCR Decision Mate V



Abb. 1.3: Triumph-Adler PC

kontrolliert, die sich während der Ausführung im Arbeitsspeicher des Computers befinden. Der Arbeitsspeicher hat die Aufgabe, Informationen zu speichern; dies sind entweder Programme oder Daten. Die Größe des Arbeitsspeichers wird in Speicherworten angegeben (8-Bit Bytes für einen 8-Bit-Mikrocomputer), und zwar als Vielfaches von 1 K, wobei 1 K = 1024 Byte entspricht. Die meisten Personal Computer bieten 64 KByte. Bürocomputer mit 16-Bit-Prozessoren verfügen meist über 128 K oder 256 KByte Speicher.

Die Mehrzahl der heute eingesetzten Speicher sind flüchtige Speicher, d h. ihr Inhalt geht nach dem Abschalten des elektrischen Netzes (220 V) verloren. Das bedeutet, daß ein Programm, das ausgeführt werden soll, von der Diskette in den Arbeitsspeicher geladen werden muß. Diese Arbeit wird vom CP/M ausgeführt.

Die Plattenlaufwerke

Weil der Arbeitsspeicher des Computers (genannt RAM: Random Access Memory) flüchtig ist und der Inhalt nach dem Abschalten verloren geht, benötigt jeder Computer Medien, die auf Dauer Informationen speichern können. Bei Kleinrechnern werden hierzu vorwiegend Plattenlaufwerke – flexible Magnetplatte (Floppy, Diskette) oder Festplatte („Hard-Disk“, Winchester-Laufwerke) – eingesetzt. Die gesamten Informationen wie Programme, sonstige Dateien mit Datenmengen oder Texten sowie eine Kopie des CP/M-Betriebssystems können auf diesen Medien auf Dauer gespeichert werden.

Der Bildschirm und die Tastatur

Das Bildschirm-Terminal (CRT-Terminal; CRT = Cathode Ray Tube) besteht aus einer Kombination von Bildschirm (ähnlich einem Fernsehbildschirm) und einer Tastatur. Mit Hilfe des Bildschirm-Terminals kann ein Computerbenutzer direkt mit dem Computer kommunizieren. Der Bildschirm gibt Informationen an den Benutzer aus. Der Bildschirmspeicher ist genauso wie der Arbeitsspeicher des Computers flüchtig und verliert seine Informationen nach Abschalten der Stromversorgung.

In vielen kommerziell genutzten Systemen wird ein konventionelles Terminal eingesetzt, das Tastatur und Bildschirm als Kombination aufweist. In den Fällen, wo die Tastatur in das Gehäuse integriert ist, wird ein separater (oder integrierter) Video-Monitor eingesetzt.

Der Drucker

Der Drucker ist eine sogenannte „Hard-Copy-Einheit“. Seine Aufgabe liegt darin, alle vom Benutzer angefragten Informationen in dauerhafter und lesbarer Form verfügbar zu machen. So wird der Drucker zum Listen von Programmen und für die Dokumentation eingesetzt.

Nachdem die wesentlichen Hardware-Komponenten dargestellt wurden, werden im folgenden die Software-Komponenten vorgestellt.

Die Software-Komponenten

Der Begriff „Software-Komponenten“ bezieht sich sowohl auf Programme als auch auf Daten. Ein Programm ist eine Folge von Befehlen, die, wenn sie sich im Arbeitsspeicher befinden, den Computer zu bestimmten Aktionen veranlassen. Daten sind Zusammenstellungen von Buchstaben, Ziffern und Sonderzeichen, die von einem Programm verarbeitet werden können. Programme und Daten werden vom Anwender mit einem Namen versehen und als Dateien bezeichnet. Im folgenden wird gezeigt, wie man bestimmte Programme einsetzt und wie verschiedene Arten von Dateien generiert und verändert werden können.

CP/M selbst ist ein spezielles Programm bzw. eine Sammlung von Programmen, die im allgemeinen auf einer Diskette gespeichert sind. Genauso sind die in diesem Buch besprochenen Programme auf Disketten verfügbar.

Es werden zwei wesentliche Softwaregruppen unterschieden: die Systemsoftware und die Anwendungssoftware. Die Systemsoftware umfaßt die gesamte Software, die im allgemeinen vom Computersystem für die notwendigen Systemarbeiten benötigt wird. Sie schließt CP/M mit einer Reihe von Hilfsprogrammen („utility programs“, Dienstprogramme) ein wie z.B. PIP und ED, auf die später eingegangen wird.

Die Anwendungssoftware umfaßt eine Sammlung von Programmen, die ein Benutzer für seine speziellen Anwendungsaufgaben einsetzt. Beispiele für Anwendungssoftware sind Programme für die Verwaltung von Adressen, für die Lagerhaltung, für die Finanzbuchhaltung oder für die Textverarbeitung.

Abgrenzung von CP/M und MP/M

CP/M ist die Abkürzung für „Control Program for Microprocessors“. MP/M ist die Abkürzung für „Multiprogramming Control Program for Microprocessors“. CP/M und MP/M sind Betriebssysteme, die bestimmte Systembefehle des Anwenders ausführen und so eine relativ einfache Nutzung aller dem Computer zugänglichen Hardware-Komponenten ermöglichen. So wird z.B. die Übertragung von Texten zum Drucker, das Lesen und die Verwendung von Informationen von der Tastatur und das Senden von Informationen zum Bildschirm organisiert. Darüber hinaus verwaltet CP/M alle internen und externen Einheiten, unter anderem auch die verfügbaren Speicherkapazitäten der Disketten und des Arbeitsspeichers.

Unter dem Sammelbegriff CP/M werden hier für die 8-Bit-Systeme das CP/M 2.2 (auch CP/M-80 genannt; für die Prozessoren 8080, 8085 und

Z80) und das neue Betriebssystem CP/M-Plus (auch CP/M 3.0 genannt), das aufwärts verträglich zu CP/M 2.2 ist, verstanden. Für die 16-Bit-Systeme wird das CP/M-86 (für die Prozessoren 8086 und 8088), das CP/M-68K (für den Prozessor 68000) und das Concurrent CP/M-86 (auch CCP/M-86 genannt) angeboten. Auf allen CP/M-Systemen kann gleichzeitig nur 1 Benutzer arbeiten (single user). Dieser kann jeweils nur eine Arbeit, wie beispielsweise editieren, drucken oder kopieren, ausführen. Lediglich mit CCP/M-86 und selbstverständlich mit MP/M kann ein Benutzer mehrere Aufgaben gleichzeitig ausführen lassen, wie z.B. editieren und drucken (multi tasking).

Einmal in den Arbeitsspeicher geladen, bildet CP/M den integrierenden Teil des Gesamtsystems und wird häufig als „das System" bezeichnet (hier muß beachtet werden, daß in der Computerfachsprache mit „System" auch vielfach die Zusammenfassung der Hardware-Komponenten wie z.B. Zentraleinheit, Druckereinheit, Bildschirm und Diskettenlaufwerke bezeichnet wird). Im folgenden soll unter „System" immer das CP/M-Betriebssystem verstanden werden.

System-Befehle

Die Arbeitsweise des Gesamtsystems wird sehr schnell klar, wenn man Schritt für Schritt die einzelnen Befehle ausführt. Das CP/M-Betriebssystem hat die wesentliche Aufgabe, dem Benutzer fortwährend alle Systemfunktionen zur Verfügung zu stellen. Nachdem der Computer eingeschaltet und das Betriebssystem in den Arbeitsspeicher geladen worden ist, fragt CP/M die Tastatur nach Befehlen ab. Dem Benutzer wird dadurch ermöglicht, in einen Dialog mit CP/M zu treten und die gewünschten Anwendungsprogramme zu starten. Sobald das Anwendungsprogramm beendet ist, übernimmt CP/M wieder die Steuerung und wartet auf den nächsten Systembefehl des Anwenders. CP/M kann so als ein ständig verfügbares Dienstprogramm angesehen werden, das immer dann zu neuen Aufgaben bereit ist, wenn kein Anwendungsprogramm abläuft. Ist ein Anwendungsprogramm einmal gestartet worden (z.B. ein Adressenverwaltungsprogramm), so befindet sich dieses im Arbeitsspeicher, und der folgende Dialog zwischen Rechner und Anwender wird von dem Anwendungsprogramm abgewickelt. Wird das Anwendungsprogramm jedoch beendet oder abgebrochen, so ist CP/M wieder aktiviert und zur Annahme neuer Befehle bereit.

Insgesamt besteht CP/M aus einer Sammlung von Programmen, die auf einer Diskette (Systemdiskette) gespeichert sind und von dort in den Arbeitsspeicher geladen werden können. Der im Computer vor dem Laden des CP/M-Grundprogramms bereits resident verfügbare Monitor oder „Bootstrap Loader" (in jedem Computer vorhanden) lädt nach dem Einschalten des Computers im allgemeinen automatisch das CP/M-Grundprogramm in den Arbeitsspeicher und startet dieses (bei Abweichungen vgl. entsprechendes Herstellerhandbuch).

CP/M bietet spezielle Befehle bzw. Routinen für die Übertragung von Informationen zwischen den am Computer angeschlossenen Geräteeinheiten, für das Starten von Programmen und für die einfache Generierung und Veränderung von Dateien. Ebenso wie jedes andere leistungsstarke Betriebssystem unterstützt CP/M viele zusätzliche Funktionen. Die wichtigsten Funktionen werden in diesem Kapitel, alle weitergehenden Funktionen in den folgenden Kapiteln beschrieben und erläutert.

CP/M, MP/M und andere Betriebssysteme

CP/M und MP/M

Im Laufe der Entwicklung wurden verschiedene Versionen von CP/M herausgegeben. In diesem Buch werden zunächst die Standardfunktionen der CP/M-Version 2.2 bzw. CP/M-86 und danach jeweils die Zusatzfunktionen von CP/M-Plus und CCP/M-86 und der MP/M-Version II beschrieben. Einige darüber hinausgehende Versionen von CP/M wurden von anderen Herstellern als „CP/M-Enhancements" entwickelt. So wurde von Cromemco beispielsweise CDOS, ein „CP/M-verträgliches" Betriebssystem mit zusätzlichen Funktionen auf den Markt gebracht. Alle in diesem Buch beschriebenen Funktionen sind bei den genannten Versionen verfügbar.

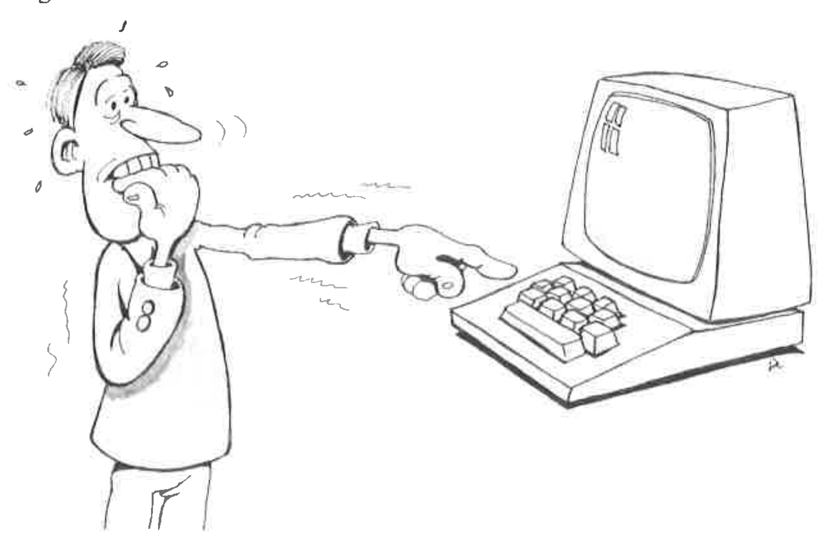
Der wesentliche Unterschied zwischen CP/M und MP/M besteht darin, daß CP/M als Einbenutzersystem entwickelt wurde. MP/M hingegen ist ein Mehrbenutzersystem, das erlaubt, den Computer über mehrere Terminals gleichzeitig zu benutzen. MP/M enthält alle CP/M- und einige zusätzliche Funktionen. Die zusätzlichen Funktionen und Leistungen von MP/M werden in jedem Kapitel noch detailliert beschrieben.

Cromemco CDOS

Das Betriebssystem CDOS von Cromemco und das MOS von Mostek erheben den Anspruch, mit der CP/M-Version 1.3 kompatibel zu sein. Das heißt, die Befehle der CP/M-Version 1.3 sind in CDOS als Untermenge enthalten. Das Gegenteil ist jedoch nicht der Fall. Programme, die CDOS-Funktionen enthalten, laufen nicht unter CP/M, da CDOS Funktionen enthält, die bei CP/M nicht verfügbar sind. CDOS benutzt ein Dateisystem, das mit dem von CP/M identisch ist, so daß jede Diskette, die von CP/M gelesen werden kann, auch von CDOS lesbar ist. Einige Unterschiede sind jedoch anzumerken. Die Bereitschaftsmeldung (prompt) des CDOS-Systems ist der Punkt „." anstelle des Zeichens „>" von CP/M. Im CDOS wird eine andere Version des PIP-Programms mit dem Namen XFER benutzt. Es arbeitet ähnlich wie das CP/M-PIP, enthält jedoch Zusatzfunktionen. PIP kann jedoch auch unter CDOS benutzt werden.

Der für die praktische Anwendung wichtigste Unterschied ist, daß einige

Steuerzeichen (Control-Characters) von CDOS interpretiert werden, die bei CP/M keine Bedeutung haben. So wird ein von CDOS geschriebenes Programm im CP/M zwar ablaufen, ohne jedoch die Steuerfunktionen zu berücksichtigen. Die Leistungen von CP/M 2.2 und CDOS bzw. MOS sind durchaus vergleichbar. Allerdings sind nicht alle Funktionen voll verträglich.



Andere Programme

Genau genommen besteht das CP/M-Betriebssystem nur aus den Routinen, die für den Dialog mit dem Computer und für die Dateiverwaltung notwendig sind. Darüber hinaus wird die Standardversion von CP/M jedoch mit verschiedenen Dienstprogrammen wie PIP und ED (eine detaillierte Beschreibung folgt in den anschließenden Kapiteln) geliefert.

Natürlich will jeder Benutzer eines Computersystems eine Vielzahl spezieller Anwendungsprogramme ausführen lassen. Zur Verdeutlichung des Ablaufs solcher Programme unter CP/M werden spezielle Beispiele mit wichtigen Definitionen erläutert.

Da die meisten Anwendungen eine bestimmte Dateiorganisation voraussetzen, muß daran erinnert werden, daß die Programme, die auf Ihrem System laufen sollen, CP/M-verträglich sein müssen. Das bedeutet beispielsweise, daß für ein bestimmtes in BASIC geschriebenes Programm ein CP/M-verträglicher Sprachinterpretierer verfügbar sein muß.

Bis hierher wurden die wichtigsten Definitionen vorgestellt. Der Computer kann nun eingeschaltet werden, damit wir mit ihm über CP/M in einen Dialog treten können.

START VON CP/M

Erster Kontakt mit dem Computer

Die Hemmschwelle vor dem Computer überwindet man am besten, indem man lernt, ihn ein- bzw. auszuschalten, ohne Schaden anzurichten. Ist der Computer ordnungsgemäß eingeschaltet, so wird das Betriebssystem die Steuerung übernehmen und auf die Eingabe von Befehlen warten (Eingabe von Befehlen oder von Abfragen). Ist die Eingabe unvollständig oder fehlerhaft, so wird das Betriebssystem eine Fehlermeldung generieren und auf neue Eingaben warten.

Wenn der Computer bereits eingeschaltet ist, sollte ein Versuch gestartet werden. Durch Eingabe beliebiger Zeichen oder Wörter mit anschließender Betätigung der „Return-Taste" kann das System zu einer Antwort veranlaßt werden. Wahrscheinlich werden die eingegebenen Zeichen mit einem zusätzlichen Fragezeichen wiederholt. Das System wartet dann auf eine neue Eingabe. Mit diesen Versuchen kann das Betriebssystem nicht zerstört werden. Allerdings können bei weiteren Fehlversuchen unter Umständen Dateien gelöscht werden. Es empfiehlt sich daher, zunächst mit der Ausführung weiterer Versuche zu warten und weiterzulesen.

Zum Starten des Systems durch Laden des CP/M-Betriebssystems wird eine Diskette (dies gilt nicht für Systeme mit einer Festplatte) benötigt. Daher folgen zunächst einige Details über Disketten.

Disketten

Heute sind zwei Gruppen von Plattenspeichern gebräuchlich: Floppy-Disks (Flexible Platten = Disketten) und Hard-Disks (Festplatten, auch

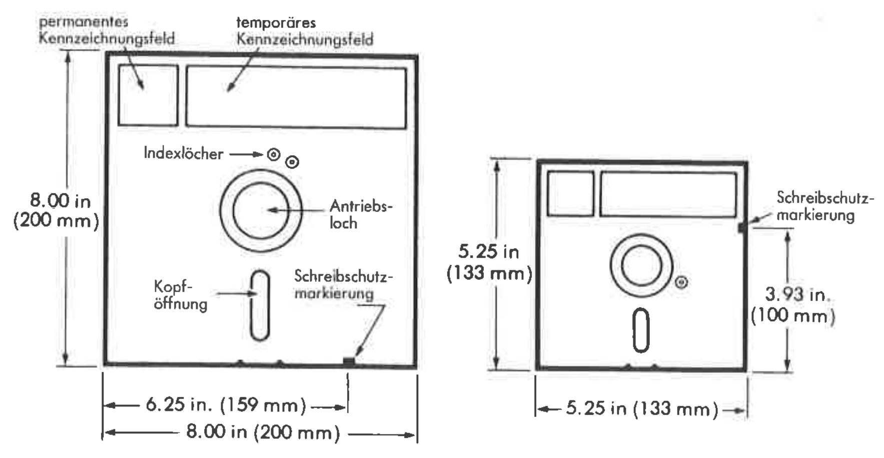


Abb. 1.4: Abmessungen von Standard- und Minidisketten

Winchesterplatten genannt oder Wechselplatten). Disketten sind in zwei Größen verfügbar: 8 Zoll und 5 1/4 Zoll. Die Abb. 1.4 und 1.5 zeigen diese preiswerten Massenspeicher. Der Nachteil dieser Datenträger liegt darin, daß sie trotz ihrer hohen Speicherkapazität sehr langsam sind und für einige Anwendungen zu wenig Speicherplatz aufweisen (z.B. große kommerziell eingesetzte Dateien). Festplatten lösen diese Probleme weitgehend. Sie bieten eine große Kapazität bei schnellem Zugriff, jedoch verbunden mit höheren Kosten. Die meisten kleineren Computersysteme sind mit einem oder beiden dieser Plattentypen ausgerüstet. Weil Disketten heute noch am gebräuchlichsten sind, beziehen sich die Ausführungen in diesem Buch vorwiegend auf diese Datenträger.

Disketten sind magnetische Datenträger und gegen äußere Einflüsse empfindlich. Sie sollten vor Magnetfeldern geschützt und vorsichtig behandelt werden. Abb. 1.6 zeigt eine Diskette.

Die quadratische Schutzhülle (vgl. Abb. 1.6) enthält eine flexible Scheibe aus Mylar-Kunststoff, die mit einer magnetisierbaren Oxydschicht beschichtet ist. Wenn auf die Diskette in einem Laufwerk zugegriffen wird, bewegt sie sich mit einer bestimmten Geschwindigkeit in der Kunststoffhülle. Am Mittelloch wird die Diskette über eine Spindelhalterung vom Motor des Laufwerks angetrieben. Über die längliche Öffnung (vgl. Abb. 1.6) in der äußeren Hülle berührt der Schreib/Lese-Kopf direkt die

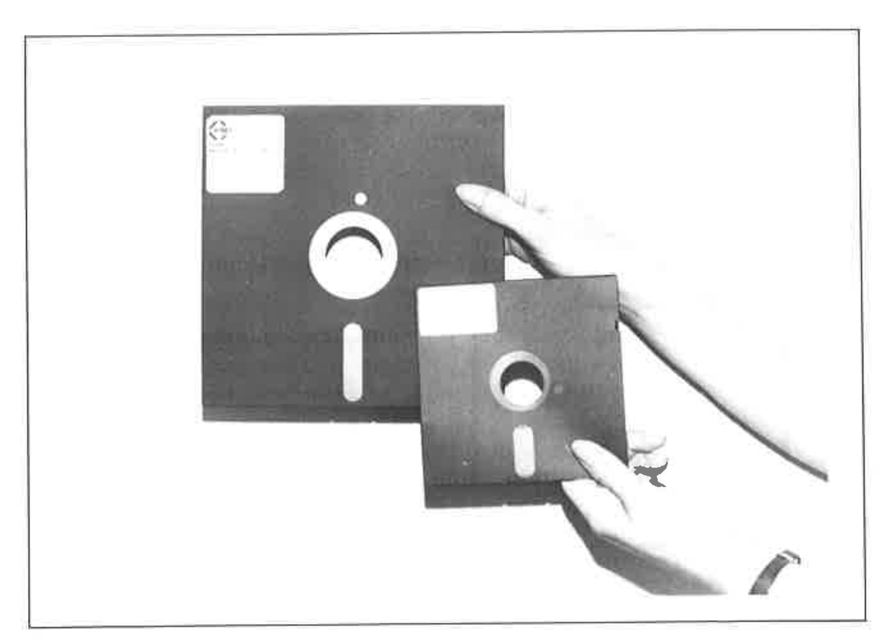


Abb. 1.5: Vergleich einer Standard- und einer Mini-Diskette

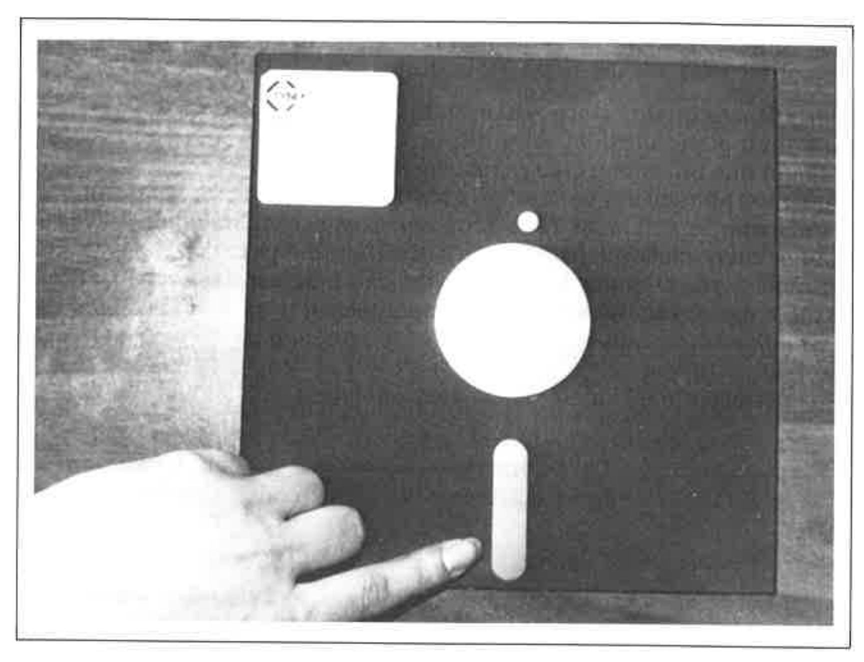


Abb. 1.6: Der Schlitz ermöglicht dem Schreib/Lese-Kopf den direkten Kontakt zur Diskette

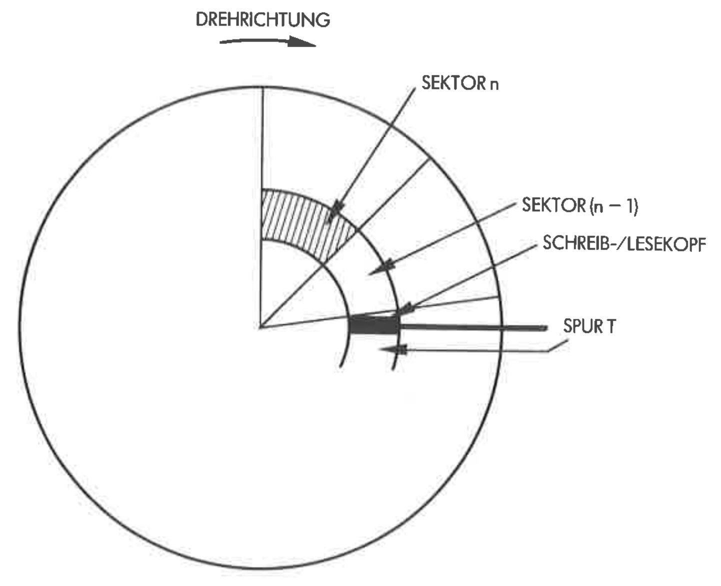


Abb. 1.7: Spuren und Sektoren

Diskettenoberfläche und ermöglicht so ein Schreiben und Lesen von Informationen auf die gleiche Weise wie bei einem Kassettenrecorder. Die Informationen sind längs konzentrischer Spuren (tracks) aufgezeichnet. Jede Spur wird von CP/M in bestimmte logische Sektoren unterteilt (vgl. Abb. 1.7).

Disketten werden meist - jedoch nicht immer - mit einem Schreibschutzloch (write-protect notch) geliefert. Bei einer 8-Zoll-Standarddiskette ist dieses Loch mit einem Papierstreifen überklebt. Wenn der Klebestreifen entfernt wird, liegt das Loch frei, und das Diskettenlaufwerk kann die Diskette nicht mehr beschreiben (Schreibschutz).

Minidisketten (5 1/4 Zoll) werden auf umgekehrte Weise vor Beschreiben geschützt. Hier muß der Papierstreifen entfernt werden, wenn die Diskette beschrieben werden soll. Wird das Loch bedeckt, so kann die Diskette nur gelesen werden. Durch diese mechanische Schreibschutzmöglichkeit können wichtige Informationen geschützt werden. So werden beispielsweise Original-Systemdisketten schreibgeschützt geliefert.

Behandlung von Disketten

Disketten müssen immer mit äußerster Vorsicht behandelt werden. So darf die Magnetschicht nicht berührt, die Diskette nicht in eine staubrei-

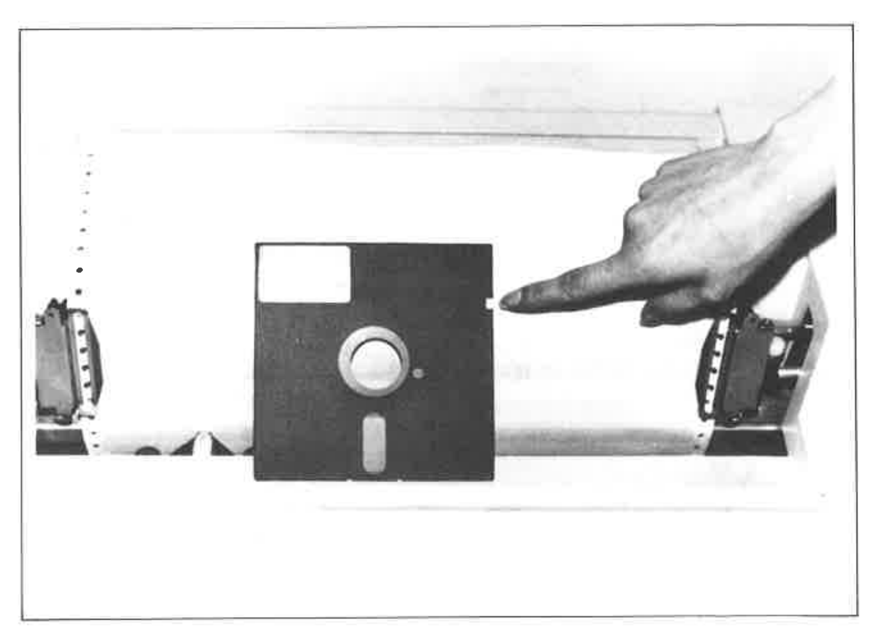


Abb. 1.8: 5 1/4 Zoll-Diskette mit Schreibschutzloch (Write-Protect Notch)

che Umgebung gebracht und nicht geknickt oder beschrieben werden. Ebenso dürfen keine magnetischen Objekte in ihre Nähe gebracht werden (Schraubenzieher und Telefongeräte), weil sie dauerhaft die Informationen auf der Diskette zerstören können. Klebeetiketten, auf denen der Inhalt der Diskette angegeben wird, sollten unbedingt vor dem Aufkleben beschriftet werden. Es ist sehr wichtig, sich von Beginn an eine Methode zu überlegen, wie auf die schonendste Weise eine Diskette in das gewünschte Laufwerk eingeschoben werden kann. Im allgemeinen ist dies möglich, wenn die Diskette an der oberen Ecke (Marken-Etikett, permanent label) mit Daumen und Zeigefinger gehalten und so in das Laufwerk eingeschoben wird (vgl. Abb. 1.9). Für das erstmalige Arbeiten mit dem Diskettenlaufwerk sollte unbedingt eine Kopie der Systemdiskette benutzt werden (für den Fall, daß sie zerstört wird).

Wenn der Mikrocomputer (oder bei separaten Laufwerken die Laufwerke) abgeschaltet wird, während die Disketten noch vollständig in den Laufwerken eingeschoben sind, so sind diese Disketten vielfach nicht mehr benutzbar. Beim Abschalten werden nämlich u.U. hohe Span-

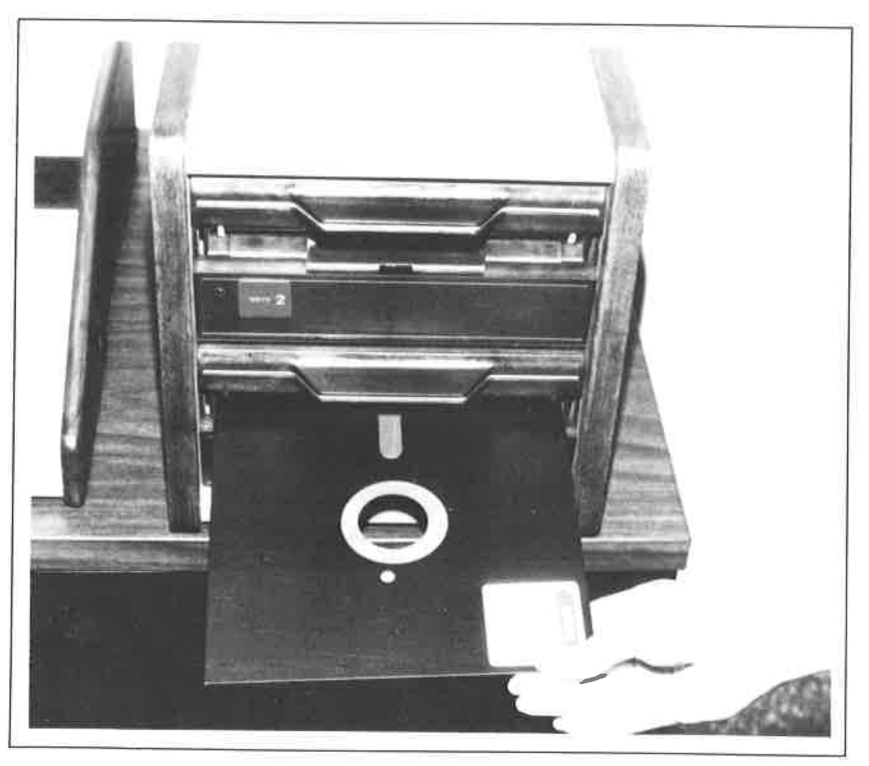


Abb. 1.9: Einschieben einer Diskette in Laufwerk A

nungsspitzen frei, die dann über die Laufwerkselektronik und die Schreib/Lese-Köpfe undefinierte Informationen auf die Disketten schreiben und damit andere Informationen zerstören. Aus diesem Grunde sollten Disketten nur bei eingeschaltetem Rechner bzw. Laufwerk in die Laufwerke eingeschoben oder aus diesen herausgenommen werden.

Im folgenden soll nun auf die einzelnen Schritte eingegangen werden, die für das Starten des Systems notwendig sind. Mit „System" wird, wie bereits erläutert, in diesem Buch immer eines der Betriebssysteme CP/M oder MP/M bezeichnet. Diese Versionen haben eine ähnliche Struktur, jedoch mit aufsteigender Leistungsbreite. Beide Systeme werden in diesem Buch beschrieben. Bezieht sich ein Befehl auf nur eines der Systeme, so wird dies ausdrücklich angegeben.



Der Ablauf

Die Systemdiskette ist eine spezielle Diskette, die das Betriebssystem CP/M (oder MP/M) enthält. Verfügt man nur über die Original-Systemdiskette, so sollte eine Kopie für die tägliche Arbeit erstellt werden.

Ist man darauf angewiesen, selbst eine Kopie zu erstellen, so sollte dieser Abschnitt bis zum Ende gelesen werden, da er eine genaue Vorgehensweise zum Starten des Systems enthält. Ist dies erfolgt, sollte man nach Kapitel 3 bzw. nach dem im folgenden beschriebenen Ablauf verfahren. Zu beachten ist folgendes: Die auf den Bildschirmausschnitten **fett** gesetzten Wörter bzw. Zeichen müssen vom Benutzer eingegeben werden. Ein „Carriage Return" (Spezialtaste der Tastatur, die dem Wagenrücklauf bei einer Schreibmaschine entspricht) wird als Pfeil angedeutet „↵". Der gesamte Ablauf im Überblick:

1. Einschieben der Systemdiskette in Laufwerk A;
2. Einschieben einer unbeschriebenen Diskette in Laufwerk B;
3. Eingabe der folgenden auf dem Bildschirmausschnitt gezeigten Zeichen:

A>**SYSGEN**↵
SYSGEN VER 2.0
SOURCE DRIVE NAME (OR RETURN TO SKIP)**A**
SOURCE ON A, THEN TYPE RETURN↵
FUNCTION COMPLETE
DESTINATION DRIVE NAME (OR RETURN TO REBOOT)**B**
DESTINATION B; THEN TYPE RETURN↵
FUNCTION COMPLETE
DESTINATION DRIVE NAME (OR RETURN TO REBOOT)↵
A>**PIP B: = A:*.*[V]**↵
(Meldungen des Systems über erfolgte Kopieroperationen)
A>

4. Herausnehmen der Kopie aus Laufwerk B, Beschriftung der Diskette auf dem Klebeetikett und Einschieben dieser Diskette in Laufwerk A.

Bei einigen Personal Computern darf für das Kopieren des Betriebssystems nicht das Programm SYSGEN verwendet werden. Der Hersteller liefert für diese Aufgabe ein spezielles Dienstprogramm, dessen Handhabung im zugehörigen Manual beschrieben ist. Das Kopieren der übrigen Dateien kann auch in diesem Fall mit PIP B: = A:*.*[V] erfolgen.

Nun kann die Kopie der Systemdiskette benutzt und erprobt werden, wie das System ein- bzw. auszuschalten ist. Da einige Computer über unterschiedliche Arten zur Einschaltung des Systems verfügen, sollten die entsprechenden Schritte gemäß den Anweisungen durchgeführt werden.

Um CP/M betriebsbereit zu setzen (Hochfahren des Systems), sind folgende Schritte notwendig:

1. Einschalten des Computers und der Peripheriegeräte;

2. Laden des CP/M-Programms von der Diskette in den Arbeitsspeicherbereich des Computers.

Die exakte Verfahrensweise ist bei jedem Computer leider unterschiedlich. Computer, die von der Konzeption her bereits für CP/M entwickelt wurden und die mit einem eigenen Monitor geliefert werden, benötigen hierzu nur einen Schritt. Das im Computer residente Monitorprogramm lädt dann CP/M automatisch von der Diskette in den Arbeitsspeicher.

Im folgenden soll für die beiden Fälle mit Hilfe eines Beispiels gezeigt werden, wie die verschiedenen Mikrocomputersysteme gestartet werden.

Einschalten eines CP/M-Systems

Die Handgriffe beim Einschalten, d.h. beim Start eines CP/M-Systems, sind heute bei den meisten Personal Computern sehr ähnlich. Auf jeden Fall sollten Sie vor der ersten Inbetriebnahme auch die betreffenden Abschnitte des zugehörigen Manuals Ihres Personal Computers sorgfältig lesen. Als Beispiel wurde hier der NCR Decision Mate V (kurz: NCR DM V) gewählt.

Der NCR DM V wird mit dem Netzschalter eingeschaltet (rechts auf der Vorderseite des Gehäuses). Danach leuchtet eine grüne Anzeigelampe auf. Gegebenenfalls muß auch der Drucker eingeschaltet werden. Einige Systeme wie z.B. der Triumph-Adler PC besitzen einen getrennten Schalter für den Bildschirm-Monitor. U.U. sind noch weitere Einheiten (Floppy- oder Winchester-Laufwerke in getrennten Gehäusen o.ä.) einzuschalten. Empfehlenswert ist deshalb die Verwendung einer Steckdosenleiste mit Netzschalter. In diesem Fall bleiben grundsätzlich alle Netzschalter auf „Ein“. Für das Ein- und Ausschalten des Personal Computers braucht dann nur der Schalter an der Steckdosenleiste betätigt zu werden.

Nun wird die Systemdiskette in das Laufwerk A eingeschoben und der Arretierungshebel um 90 Grad nach rechts gedreht (siehe auch Abschnitt „Behandlung von Disketten“ und Abb. 1.9). Laufwerk A ist das linke mit dem Buchstaben A gekennzeichnete Laufwerk. Das Label der Diskette zeigt nach links (vgl. Abb. 1.10). Falls eine CP/M 2.2 System-Diskette verwendet wurde, erscheint nach einigen Sekunden auf dem Bildschirm folgende Meldung:

```
CP/M (R) 2.2 for NCR DECISION MATE V
D 006-0051-1000
Copyright © 1982, DIGITAL RESEARCH
Serial Number xxxxx

A>
```

Wird CP/M-86 benutzt (falls der DM V zusätzlich den 8088-Prozessor enthält), sieht die Meldung auf dem Bildschirm ähnlich aus.

Haben Sie den NCR DM V falsch bedient, so erscheint statt der „Erfolgsmeldung“ eine Fehlermeldung auf der letzten Zeile des Bildschirms. Im Fall von

A: MOUNT O.S. DISK (CR)

haben Sie versehentlich eine Floppy im Laufwerk A eingesetzt, die kein CP/M-System auf den Systemspuren enthält. Entfernen Sie diese wieder, setzen eine Systemdiskette ein und betätigen Sie die Return-Taste.

Meldet der NCR DM V

DISK A: NOT READY (CR)

so haben Sie vergessen, die Systemdiskette ordnungsgemäß ins Laufwerk einzusetzen (z.B. wurde der Arretierungshebel nicht gedreht, oder die Floppy wurde verkehrt eingeschoben). Diese Meldung erscheint auch, falls Sie nach dem Einschalten des Netzes zu lange mit dem Einsetzen der Floppy gewartet haben. In diesem Fall brauchen Sie nur die Return-Taste zu betätigen.

Hat Ihr Personal Computer ein Winchester-Laufwerk, so wird das CP/M-Betriebssystem automatisch nach dem Einschalten von diesem geladen, und die obige Bereitschaftsmeldung erscheint ohne weitere Handgriffe.

Einige wenige Systeme wie z.B. der „CE Makro Computer“ (Basis: Mostek Combo Platine mit dem Z80A Prozessor) laden das vollständige CP/M-System beim „Kaltstart“ nicht von der Floppy, sondern von einem Nur-Lese-Speicher (EPROM) in den Arbeitsspeicher (RAM; 64 kByte). Durch einen zusätzlichen Leseversuch auf der Floppy wird lediglich festgestellt, welcher Typ einer Floppy sich im Laufwerk A befindet (einseitig oder zweiseitig, einfache oder doppelte Schreibdichte).

Ein anderes Beispiel für einen Sonderfall bietet der Sirius 1: Verfügt das System über zwei Floppy-Laufwerke (also keine Festplatte), so meldet sich CP/M-86 nach dem Kaltstart mit:

```
Sirius 1 Mikrocomputer
KONFIGURATION
Betriebssystem   CP/M-86 1.1
E/A System       Version 2.4
Tastatur         Deutsch 02a
Zeichensatz      Deutsch 02

A>
```

Dies entspricht weitgehend dem Standard. Für den Sirius 1 mit einem Floppy- und einem Winchesterlaufwerk ist zur Zeit jedoch nur das Betriebssystem MS-DOS von Microsoft verfügbar. In diesem Fall muß zunächst der Kaltstart (in gleicher Weise) für MS-DOS ausgeführt werden. Mit Hilfe des mitgelieferten Programms CPM.CMD können jedoch CP/M-Programme aufgerufen und ausgeführt werden, d.h. CP/M-86 wird „emuliert“ (nachgebildet). Der Aufruf des CP/M-86-Programms FAKTURA.CMD erfolgt von MS-DOS aus mit

CPM FAKTURA↵

Auf diese Weise können nicht nur CP/M-Anwenderprogramme, sondern auch Standardsoftware wie WordStar, das Datenbanksystem dBASE oder InfoStar als CP/M-86-Programme (nach einer Erweiterung des Sirius 1 mit einer Festplatte) ohne Änderungen genutzt werden.

ANWENDUNG VON CP/M

Startbereit

Die bisher beschriebenen Befehle veranlaßten eine sogenannte „bootstrap operation“. Andere Bezeichnungen sind Kaltstart („cold start“ oder „cold boot“). Mit der Bezeichnung Kaltstart ist nicht gemeint, daß der Rechner bis vor der Initialisierung eines Kaltstarts noch nicht eingeschaltet war und somit „kalt“ ist. Vielmehr ist damit der Systemstart vom niedrigsten Systemzustand aus gemeint. Mit dem Ausdruck „bootstrap“ („sich selbst an den Haaren hochziehen“) wird die Grundoperation bezeichnet, mit der der Monitor CP/M von der Diskette lädt und CP/M sich danach selbst startet. Der Unterschied zwischen einem Kaltstart und einem Warmstart wird noch erläutert. Im folgenden soll erklärt werden, was unter „A“ (oder „0A“ z.B. im Fall von MP/M) verstanden wird und was ein „prompt“ ist.

Systembereitschaftsmeldungen (System prompts)

Ein „prompt“ ist eine Mitteilung oder ein Symbol, das vom System immer dann gesendet wird, wenn das System für die nächste Eingabe von Befehlen bereit ist. Jedes System hat ein solches „prompt“. CP/M verwendet das Symbol „A>“. MP/M und CCP/M benutzen „0A>“. Der Buchstabe A steht für das Laufwerk A,und 0 steht für den Benutzer 0. Die Benutzerbereiche werden in Kapitel 2 näher beschrieben. Da vorerst keine Änderungen der Benutzerbereiche vorgenommen werden, sind diese Informationen jetzt noch nicht notwendig.

Das Systembereitschaftszeichen teilt jeweils mit, welches Disketten- oder Plattenlaufwerk als Arbeitslaufwerk (default drive) gerade eingeschaltet

ist. Jedes System verfügt über mindestens ein Laufwerk, das mit „A" gekennzeichnet ist. Weitere Laufwerke werden mit „B", „C" usw. bezeichnet. Unter der Voraussetzung, daß an das System zwei Laufwerke angeschlossen sind, kann mit dem folgenden Befehl auf Laufwerk B umgeschaltet werden:

```
A>B:↵
```

Die Antwort lautet:

```
B>
```

Das Arbeitslaufwerk ist nun das Laufwerk B, was an der Bereitschaftsmeldung

```
B>
```

erkennbar ist. Im folgenden wird also das Laufwerk B als „selektiertes" Laufwerk verwendet, falls in Verbindung mit einem Dateinamen kein anderes angegeben wird. Auf Laufwerk A kann mit dem Befehl

```
B>A:↵
A>
```

zurückgeschaltet werden.

Dateien (files)

Disketten speichern Informationen in Dateien (files). Um diese Informationen verwenden zu können, muß dem Computer mitgeteilt werden, daß er auf die entsprechende Diskette oder Platte (über das Disketten- oder Plattenlaufwerk) zugreifen und dort eine Datei über eine Bezeichnung (Dateibezeichnung = filename) aufsuchen soll. Mit Hilfe der Systemdiskette in Laufwerk A wurde das System geladen. Falls noch nicht zu einem anderen Laufwerk umgeschaltet wurde, ist immer noch Laufwerk A zugeschaltet. In diesem Fall erhält man die Bereitschaftsmeldung „A>". Auf ein anderes Laufwerk kann nur dann geschaltet werden, wenn sich dort auch eine Diskette befindet. Hierauf wird später in diesem Kapitel noch eingegangen.

Benutzung des Tastenfeldes

Wenn nur die Return-Taste gedrückt wird, so erscheint als Antwort darauf wieder ein Bereitschaftszeichen. Betätigt man mehrere Male hintereinander die Return-Taste und beobachtet die Reaktionen auf dem Bildschirm, so kann man erkennen, wie einfach es ist, Leerzeilen zum Computer zu senden. Die Return-Taste wird gebraucht, um einen Befehl zum Computer zu senden. Ein Befehl schließt immer mit einem RETURN ab – dieser Befehl wird in diesem Buch mit dem Symbol ↵ angedeutet. Es gibt nur wenige Ausnahmen, bei denen ein Befehl nicht durch ein RETURN abgeschlossen wird, so z.B. beim Senden von Steuerzeichen (gleichzeitiges Drücken der CTRL-Taste mit einer anderen Taste). Darauf wird später noch eingegangen.

Um mit der Arbeitsweise des Systems vertraut zu werden, sollten die folgenden Beispiele ausgeführt werden:

Eingabe einer beliebigen Zeichenfolge mit abschließendem RETURN:

```
A>ABCDEXYZ↵
ABCDEXYZ?
A>
```

Sollte das System eine solche Fehlermeldung nicht auf dem Bildschirm zeigen, dann muß die CONTROL-Taste gleichzeitig mit dem Buchstaben C (vgl. Abb. 1.15) gedrückt werden. Diese Kombination (CTRL- und C-Taste) veranlaßt bei CP/M einen Warmstart (warm start) bzw. ein Nachladen des Systems (warm boot bzw. system reboot).

Ein Warmstart unterbricht die gerade ablaufende Operation und startet erneut das Betriebssystem. Es erscheint daraufhin wieder das Bereitschaftszeichen des Systems.

Falls diese Funktion nicht wie beschrieben abläuft, kann der Abschnitt „Fehlerprüfung“ in diesem Kapitel zu Rate gezogen werden.

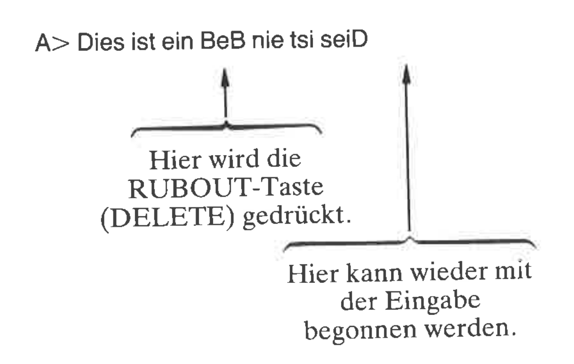
Der Gebrauch von CTRL-C sollte geübt werden (in diesem Buch steht das Zeichen „^“ als Abkürzung für CTRL; ^C entspricht also CTRL-C). Beachten muß man, daß vor und während des Drückens des C die CTRL-Taste bereits gedrückt sein muß.

Genauso sollte der Gebrauch der Taste RUBOUT (DELETE) geübt werden. Wenn irgendetwas eingegeben wurde, so kann das letzte Zeichen durch Drücken der Taste RUBOUT (DELETE) gelöscht werden. Bei verschiedenen Terminals kann durch entsprechend langes Drücken der RUBOUT-Taste die gesamte vorher eingegebene Zeile gelöscht werden.

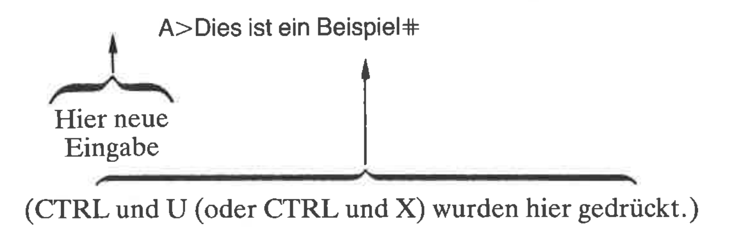


Abb. 1.10: CTRL-C initialisiert einen Warmstart

Bei einigen CP/M-Versionen wird das Zeichen, das gelöscht wurde, dupliziert und nochmals auf dem Bildschirm gezeigt (re-display, echoe):



Eine andere Möglichkeit, eine ganze Zeile zu löschen, ist das gleichzeitige Drücken der CTRL-Taste und der Taste U (Herunterdrücken und Halten der CTRL-Taste und Drücken der U-Taste). CTRL-Taste und X haben die gleiche Wirkung. Diese beiden Steuerzeichen werden mit ^U und ^X abgekürzt. Nach der Eingabe von ^U oder ^X zeigt das System als Antwort ein Doppelkreuz (#), was bedeutet, daß alle Zeichen, die links vor dem # stehen, gelöscht sind:



Danach kann wieder mit der Eingabe einer neuen Zeile begonnen werden. Ebenso kann durch Betätigung der Return-Taste eine Leerzeile zum Computer gesandt werden, worauf das System mit einem Bereitschaftszeichen antworten wird.

Fehlerprüfung

Es kann vorkommen, daß etwas Ungewöhnliches geschieht, ohne daß man die Ursache direkt ermitteln kann. Wenn die Return-Taste gedrückt wird, ohne daß eine Antwort vom Rechner kommt, so ist der Computer gerade mit einer Aufgabe beschäftigt (z.B. Ausführung eines Programms). Falls zufälligerweise der Name eines Programms eingegeben wurde, das nicht gestartet werden sollte, so kann man einen Abbruch durch ^C erzwingen (Warmstart).

Wenn ein Warmstart nicht zum Bereitschaftszeichen des Systems führt, sollte überprüft werden, ob eine Kontrolleuchte eines Disketten-Laufwerks leuchtet. Das Leuchten dieser Kontrollampe zeigt, daß der Rechner gerade von der Diskette liest. Wenn die Kontrollampe leuchtet, obwohl sich keine Diskette in dem Laufwerk befindet, kann eine Diskette in das Laufwerk geschoben werden, damit der Rechner den Lesevorgang starten und beenden kann. Führt auch dies nicht zu einer Bereitschaftsmeldung, so muß noch einmal ein Kaltstart vorgenommen werden (vgl. Abschnitt „Einschalten, Disketten einschieben und Laden“ am Beginn dieses Kapitels).

Systemdiskette

Die Systemdiskette muß jeweils an das verwendete System angepaßt sein. Wenn irgendein Hardwareelement im System ausgewechselt wurde, wird die Originalsystemdiskette u.U. nicht mehr funktionieren.

Wenn z.B. das Bildschirm-Terminal, Floppy-Laufwerke oder andere Geräte aus dem System herausgenommen und durch andere ersetzt werden, so wird auch u.U. eine andere Systemdiskette benötigt. Falls man die Hardwarekonfiguration von Zeit zu Zeit verändert, ist es empfehlenswert, die einzelnen Systemdisketten sorgfältig zu beschriften, um Verwechslungen zu vermeiden.

Prüfen des Inhaltsverzeichnisses

Die Systemdiskette befindet sich in Laufwerk A. Sie enthält CP/M sowie weitere Dateien. Um dies zu überprüfen, muß der Befehl DIR (directory) eingegeben werden:

```
A>DIR↵
A:PIP    COM : ASM     COM : LOAD    COM : PROGR COM
A:STAT   COM : PROGR   INT
```

Listet der Bildschirm die Information zu schnell auf, so kann durch Drükken der beiden Tasten CTRL und S eine Unterbrechung der Auflistung erreicht werden. Das Drücken von CTRL und Q (bei CP/M-80 auch andere Tasten) setzt das Auflisten fort.

Das oben aufgeführte Beispiel zeigt die Anwendung des DIRectory-Befehls. Jede Diskette (oder Platte) hat ein Inhaltsverzeichnis mit den Dateibezeichnungen für jede einzelne Datei, die sich auf der Diskette (Platte) befindet. Weil Laufwerk A als Arbeitslaufwerk geschaltet ist und sich die Systemdiskette in Laufwerk A befindet, müssen die oben gezeigten Dateien auf der Systemdiskette vorhanden sein. Jeder Dateibezeichnung ist die Zeichenfolge „A:“ vorausgestellt, was anzeigt, daß die Datei sich auf der Diskette in Laufwerk A befindet. Das erste Wort „PIP“ mit dem Folgewort „COM“ bildet die gesamte Dateibezeichnung „PIP.COM“. „PIP“ ist der eigentliche Dateiname und „COM“ die Dateikennung (extension), die den Dateityp kennzeichnet. Dateiname und Dateikennung werden bei der Eingabe durch einen Punkt getrennt.

Dateikennungen werden oft auch als Dateitypen bezeichnet (file types). So sind z.B. alle Dateien mit der Kennung COM sogenannte „command files“. Alle Dateien mit dem Typ BAS sind BASIC-Quellprogramme, und alle Dateien mit der Kennung INT sind BASIC-Intermediate-Programme. Die Angabe von Dateikennungen ist allerdings in einigen Fällen nicht unbedingt notwendig.

Die Dateikennungen können individuell gewählt werden, um so eine sinnvolle Strukturierung der Dateien zu erhalten. Es ist wichtig und organisatorisch notwendig, Dateien voneinander abzugrenzen. So kann z.B. die Directory (Inhaltsverzeichnis) auch mehrere Dateien mit demselben Namen, jedoch mit unterschiedlichen Kennungen enthalten.

Beispiele:

TEXT.WRK	(Arbeitsdatei - working file)
TEXT.BAK	(gesicherte Datei - backup file)

oder

PROG.BAS	(Basis-Quellprogramm)
PROG.INT	(übersetztes BASIC-Programm)

Die vollständige Dateibezeichnung wird benötigt, wenn eine allgemeine Datei angesprochen werden soll. Ausgenommen sind Befehlsdateien (command files), die später in diesem Kapitel beschrieben sind.

Zunächst soll gezeigt werden, wie eine Diskette in ein Laufwerk eingeschoben und dort eine neue Datei angelegt wird.

ABLAUF EINES PROGRAMMS

In diesem Abschnitt wird eine einfache Datei mit einem vorhandenen Programm aufgebaut und verwaltet. Es wurde dafür ein Programm ausgewählt, das eine heute typische Bedienerführung bietet. Das im folgenden kurz beschriebene Programm ADRESSEN wurde von Lifeboat, Deutschland, für

- die Verwaltung einer Adreß-Datei,
- das (selektive) Drucken eines Adreßverzeichnisses,
- das Bedrucken von Adreßaufklebern und
- die Erzeugung von Formbriefen

entwickelt. Für die Bedienerführung wird eine Menüsteuerung verwendet. Die im jeweils nächsten Bearbeitungsschritt möglichen Funktionen werden im unteren Drittel des Bildschirms angezeigt. Ein Weiterschalten, d.h. die Anwahl der nächsten Funktion, erfolgt mit Hilfe der Blank-Taste (Zwischenraum-Taste). Die jeweils angewählte nächste Funktion wird durch inverse Darstellung des betreffenden Begriffs angezeigt. Die Auslösung, d.h. der effektive Aufruf, der Funktion erfolgt durch Drükken der Return-Taste. Im folgenden wird das nun an einem einfachen Beispiel, nämlich der Eingabe einer Adresse, dargestellt.

Zunächst wird eine Diskette mit dem Programm ADRESSEN.COM in das Laufwerk A und eine andere (leere, formatierte) Diskette für die Adreß-Datei in das Laufwerk B eingesetzt. Der Aufruf des Adreßverwaltungsprogramms erfolgt durch

ADRESSEN KUNDEN↵

Dabei ist ADRESSEN der Name des Programms (ADRESSEN.COM) und KUNDEN der Name der Adreß-Datei (KUNDEN.DAT). Nach diesem Aufruf meldet sich das Programm mit:

```
-------------------------------------------
B M - A D R E S S V E R W A L T U N G
Copyright © 1984 by
J. Pfotenhauer, Mikrocomputer-Anwendungen
Version 1.01          Ser. Nr. xxxxx
-------------------------------------------
Teile dieses Programmes unterliegen dem
Copyright von Digital Research Inc.
-------------------------------------------

Auf welchem Laufwerk ist die Datei gespeichert ?
```

In unserem Fall muß als nächstes der Buchstabe B eingeben werden, da sich die Floppy für die Adreß-Datei im Laufwerk B befindet. Das Programm bestätigt diese Eingabe mit dem zusätzlichen Text:

```
Bestaetigen Sie diese Eingabe mit Return (Enter) oder geben
Sie ein anderes Zeichen ein, um ein anderes Laufwerk
zu waehlen.
```

Nach der Bestätigung mit der Return-Taste erscheint unten auf dem Bildschirm das erste Menü:

```
Auswahl: Eingaben  Suchen  Drucken  Abspeichern  Programmende
         Waehlen mit Leertaste, Ausloesen mit Return (Enter)
Status:  Dateiname:       Rekords       in Gebrauch
         B:KUNDEN.DAT     1             0
```

Als nächsten Schritt wollen wir eine Adresse eingeben. Weil der „Zeiger" bereits auf Eingaben steht, brauchen Sie nur die Return-Taste zu drükken. Es erscheint auf dem Bildschirm folgende Maske:

```
Adressenverwaltung
------------------------------------------------------------------------
Kundennummer                                        Zusatzfeld 1
Anrede
Vorname                                             Zusatzfeld 2
Nachname
zus. Name
Firma
Abteilung
Strasse
LKZ
PLZ
Ort
Telefon       (    )    /
Telex
```

Sie können jetzt die neue Adresse eingeben. Der Cursor steht auf der Kundennummer, mit der Sie beginnen müssen. Jede einzelne Eingabe wird mit der Return-Taste beendet. Nach Eingabe der Kundennummer prüft das Programm zunächst, ob diese Nummer bereits einmal auftrat. Ist diese keine Doublette, so wird das Menü ersetzt durch:

```
ESC → Eingabe/Aendern Beenden
```

```
Adressenverwaltung
------------------------------------------------------------------------
Kundennummer      4711                  Zusatzfeld 1
Anrede            Herr                  CP/M-Software
Vorname           Wolfgang
Nachname          Müller                Zusatzfeld 2
zus. Name
Firma             XY-Bürocompter
Abteilung         Software
Strasse           Bitring 256
LKZ               D
PLZ               2347
Ort               Posemuckel
Telefon           (     )0417/56 781
Telex
```

Die Eingabe oder Änderung einer Adresse wird nach Beendigung aller Angaben mit der ESC-Taste abgeschlossen. Vorher können mit Hilfe der Cursor-Taste (bzw. Cursor nach oben: ^E), der Rücktaste und der Delete-Taste auch bereits eingegebene Informationen editiert werden. Beispielsweise kann auf diese Weise vorstehende Adresse eingeben werden.

```
ESC → Eingabe/Aendern Beenden

Status:   Dateiname:       Rekords      in Gebrauch
          B:KUNDEN.DAT     1            0
```

Die Eingabe dieser Adresse wird mit der ESC-Taste beendet. Das Menü wird nun in

```
Auswahl:  Speichern   Löschen   Beenden
          Waehlen mit Leertaste, Ausloesen mit Return (Enter)

Status:   Dateiname:       Rekords      in Gebrauch
          B:KUNDEN.DAT     2            1
```

geändert. Aktivieren Sie jetzt durch die Return-Taste die Funktion „Speichern". Die Adresse wird gespeichert, und anschließend kann direkt die nächste Adresse eingegeben werden. Abgeschlossen wird die Eingabe neuer Adressen schließlich durch Anwählen der Funktion „Beenden" mit der Blank-Taste und Aufruf durch die Return-Taste. Die zuletzt eingegebene Adresse wird zusätzlich abgespeichert. Es erscheint dann wieder (z.B. nach Eingabe von 14 Adressen) das Menü:

```
Auswahl:  Eingaben  Suchen  Drucken  Abspeichern  Programmende
          Waehlen mit Leertaste, Ausloesen mit Return (Enter)
Status:   Dateiname:       Rekords      in Gebrauch
          B:KUNDEN.DAT     15           14
```

Soll nun das Programm beendet werden, so wird wieder mit der Blank-Taste „Programmende" angewählt und mit der Return-Taste aufgerufen. Das CP/M-System meldet sich nun wieder mit A> zurück. Die Adressen sind in der Datei KUNDEN.DAT auf der Floppy im Laufwerk B gespeichert.

Dieses Beispiel demonstriert, wie einfach die Benutzung eines CP/M-Programms mit einer Bedienerführung auch für den Anwender ohne Erfahrungen mit Personal Computern sein kann.

DATEIGENERIERUNG

Eine einfache Textdatei kann auch mit Hilfe des CP/M-Editors „ED“ generiert werden. Dieser Editor ist zwar gegenüber leistungsfähigeren Editoren, die für CP/M-Systeme geschrieben wurden, etwas schwierig zu handhaben, dennoch soll er an einem Beispiel erläutert werden.

Der ED-Befehl ist ein sogenannter transienter Befehl (transient command), was soviel bedeutet, daß er sich als Befehlsdatei (ED.COM) auf der Systemdiskette befindet und nur vorübergehend (während der Ausführung) in den Arbeitsspeicher geladen wird. Mit dem DIR-Befehl läßt sich überprüfen, ob sich dieser Editor auf der Systemdiskette befindet:

Eingabe:

```
B>A:↵        (Rücksprung zu Laufwerk A)
```

Eingabe:

```
A>DIR↵       (Überprüfung der Directory;
              ED.COM muß mit aufgelistet werden)
```

Eingabe:

```
A>B:↵        (Rücksprung zu Laufwerk B)
```

Wenn der Editor „ED.COM“ aufgerufen wird, wird er normalerweise vom Arbeitslaufwerk geladen. Ist er auf diesem Laufwerk nicht vorhanden, so führt das beim Standard-CP/M zur Fehlermeldung. CP/M-Plus, MP/M und CCP/M versuchen jedoch in diesem Fall, das Programm vom Laufwerk A zu laden. Hier kann auch, wenn auf Laufwerk B geschaltet ist, der Editor von A aufgerufen werden.

Bei der Eingabe eines „transient command“ darf die zugehörige Datei-

kennung „.COM“ nicht mit eingegeben werden. Zum Aufrufen des Editors ED.COM wird daher nur „ED“ (bzw. „A:ED“ zur Spezifizierung des Laufwerks) eingegeben. Wird die gesamte Bezeichnung „ED.COM“ eingegeben, so führt dies zu einer Fehlermeldung.

Zunächst ist eine Dateibezeichnung festzulegen wie z.B. „BEISPIEL.TXT“, wobei darauf geachtet werden muß, daß Dateinamen (hier: BEISPIEL) und Dateikennung (hier: TXT) mit einem Punkt voneinander getrennt werden. Die Dateikennung ist hierbei zwar nicht notwendig, erleichtert aber das spätere Auffinden dieser Datei.

Es soll nun der ED-Befehl ausgeführt werden, indem sowohl „ED“ als auch die gewählte Dateibezeichnung „BEISPIEL.TXT“, durch ein Leerzeichen voneinander getrennt, eingegeben werden.

```
B>ED BEISPIEL.TXT↵
ED?
B>
```

Achtung! Hier wurde vergessen, für den ED-Befehl das Laufwerk zu spezifizieren (ED.COM befindet sich auf Laufwerk A, während das System auf Laufwerk B geschaltet ist).

Neuer Versuch:

```
B>A:ED BEISPIEL.TXT↵
NEW FILE
*
```

Der Editor arbeitet. Hinter ED muß unbedingt die zu editierende Datei „BEISPIEL.TXT“ angegeben werden.

Das Zeichen * ist das Bereitschaftszeichen des Editors (editor's prompt). Es zeigt an, daß das Editierprogramm geladen und gestartet wurde und auf den nächsten ED-Befehl wartet. Die einzelnen ED-Befehle sind im Kapitel 4 detailliert beschrieben. Einige sollen aber hier schon behandelt werden:

Der I-Befehl (insert) fügt neuen Text in eine Datei ein, der B-Befehl (begin) setzt den Zeiger an den Anfang der Datei, und der T-Befehl (type) listet den Inhalt der Datei am Bildschirm auf. Der E-Befehl (exit) speichert die Datei auf die Diskette und beendet den Editiervorgang mit dem Editor.

Mit dem CP/M-Editor können Textdateien generiert und verändert werden. Textdateien sind Dateien, deren Inhalt durch Zeichen in einem bestimmten Binärcode, dem ASCII-Code, definiert ist. Jedem Zeichen auf der Terminaltastatur entspricht ein bestimmter ASCII-Code (vgl. Anhang C). In den wenigsten Fällen wird der Anwender von diesen Zeichen Gebrauch machen. Es ist jedoch recht nützlich, die Codierung lesen zu können, wenn z.B. plötzlich undefinierbare Zeichen und Kombinationen auftauchen.

Zur Einfügung neuer Zeichen in die generierte Datei wird zunächst der I-Befehl und dann der gewünschte Text eingegeben. Jede Zeile wird wie bei einer Schreibmaschine mit einem Wagenrücklauf (Carriage Return, RETURN oder CR) abgeschlossen. Die Einfügung von Text wird mit CTRL-Z (^Z) abgeschlossen.

```
*I↵
Das ist meine neue Textdatei↵
mit dem Namen "BEISPIEL.TXT"↵
^Z
*
```

Einfügen einer dritten Zeile:

```
Das ist Zeile drei.↵
```

Wenn der gespeicherte Text nun auf dem Bildschirm aufgelistet werden soll, so wird die folgende Zeichenfolge eingegeben:

```
*B↵        (Sprung an den Anfang der Datei)
*#T↵       (Auflisten des Dateiinhalts)
Das ist meine neue Textdatei↵
mit dem Namen "BEISPIEL.TXT"↵
Das ist Zeile drei.
*
```

Der B-Befehl setzt den Zeiger an den Anfang der Textdatei, der T-Befehl (type) listet eine Textzeile (bzw. bei der Eingabe von #T die gesamte Datei) auf. Mit Hilfe des E-Befehls kann die Datei wie folgt auf der Diskette gespeichert werden:

```
*E↵
B>
```

Der E-Befehl speichert die noch im Arbeitsspeicher befindliche Datei auf der Diskette ab und beendet das Editorprogramm. Das System meldet sich dann wieder bereit, und die Datei befindet sich auf der Diskette von Laufwerk B. Falls der E-Befehl nicht gegeben wird, gehen die bis dahin eingegebenen Informationen verloren. Deshalb sollte der E-Befehl während der Generierung größerer Texte möglichst häufig gegeben werden. Der über das Bildschirmterminal eingegebene Text wird nämlich zuerst im internen Speicher (Arbeitsspeicher) des Rechners gespeichert, wo er dann leicht modifiziert werden kann. Erst wenn der E-Befehl eingegeben wird, wird vom Arbeitsspeicherinhalt eine Kopie auf die Diskette gespeichert.

Wenn der Editor über das Abschalten des Systems oder durch einen Warmstart (^C) verlassen wird, dann ist der Arbeitsspeicher nicht mehr direkt zugänglich, und die vorher eingegebenen Informationen sind verloren. Aus Sicherheitsgründen sollte daher möglichst oft neu eingegebener Text gespeichert werden, da die Gefahr des versehentlichen Drükkens der Tasten CTRL-C sehr groß ist.

DATEIVERÄNDERUNG

Anzeige von Dateiinhalten

Wenn das Bereitschaftszeichen des Systems wieder auf dem Bildschirm erscheint, kann mit Hilfe des TYPE-Befehls und einer nachfolgenden Dateibezeichnung die gewünschte Datei aufgelistet werden:

```
B>TYPE BEISPIEL.TXT↵
Das ist meine neue Datei
mit dem Namen "BEISPIEL.TXT"
Das ist Zeile drei.
```

Unter Verwendung der oben beschriebenen ED-Befehle können bereits die ersten Dateien generiert werden. Die vollständige Beschreibung des Editors erfolgt in Kapitel 4. Es sollte jedoch darauf hingewiesen werden, daß für CP/M leistungsfähigere Editoren (oder Textverarbeitungsprogramme) mit ähnlichen oder anderen Dateistrukturen verfügbar sind. Der Anwender sollte den für seine Bedürfnisse am besten geeigneten Edi-

tor einsetzen. Einige dieser Editoren sind sehr einfach in der Bedienung, andere schwieriger, besitzen jedoch zusätzliche Möglichkeiten der Textmanipulation. Falls der Benutzer über mehrere Editoren verfügt, sollten die Dateien mit den Editoren geändert werden, mit denen sie erstellt wurden.

UMBENENNEN VON DATEIEN (RENaming files)

Der REN-Befehl wird zum Umbenennen (renaming) von Dateibezeichnungen benutzt. Er wird wie folgt angewendet:

```
REN neue Dateibezeichnung = alte Dateibezeichnung
```

Hierbei sind immer die vollständigen Dateibezeichnungen anzugeben.

Beispiel:

```
REN DATEI2.TXT = ALT.TXT
```

oder

```
REN KOLLEGEN.DAT = NAMEN.DAT
```

Nach der Ausführung des REN-Befehls existiert die Datei unter dem Namen „ALT.TXT“ nicht mehr. Sie ist jetzt nur noch unter dem Namen „DATEI2.TXT“ zu erreichen. Entsprechend hat die Datei im zweiten Beispiel die neue Dateibezeichnung „KOLLEGEN.DAT“ erhalten. Wichtig ist in jedem Fall, daß immer die vollständige Dateibezeichnung (Dateiname und, wenn vorhanden, die Dateikennung nach dem dann anzugebenden Punkt) anzugeben ist.

LADEN NEUER DISKETTEN (Anwendung eines Warmstarts)

Sehr bald nach der Erstbenutzung eines neuen Computersystems entsteht der Wunsch, Disketten zu kopieren und die Disketten in Laufwerk B gegen andere auszutauschen. Bevor dies ohne Fehler durchgeführt werden kann, muß die Wirkung eines Warmstarts analysiert werden. Wenn die Diskette eingeschoben ist, liest CP/M bei einem Warmstart automatisch die Directory in den Arbeitsspeicher ein und selektiert die Diskette. Wird die Diskette in Laufwerk B ausgetauscht und versucht man sofort danach, auf diese neue Diskette zu schreiben, so kann dies zu einer Fehlermeldung führen, da eine spezielle Routine die Diskette vor unbeabsichtigtem Beschreiben schützt.

Aus diesem Grunde muß bei jedem Wechsel einer Diskette dem Betriebssystem erst erlaubt werden, auf die neue Diskette zu schreiben. Dies wird durch einen neuen Start von CP/M (Warmstart) erreicht.

Es soll vorausgesetzt werden, daß sich die Systemdiskette in Laufwerk A befindet und eine neue Diskette in Laufwerk B eingeschoben wurde. Der Warmstart (warm boot) wird dann durch Eingabe von „^C" veranlaßt; dem Rechner wird damit ermöglicht, die neue Diskette in Laufwerk B zu beschreiben. Beim Beschreiben einer Diskette orientiert sich der Computer an einer Tabelle, die im Arbeitsspeicher gespeichert ist. Wird eine Diskette ausgetauscht, so befindet sich immer noch die alte Tabelle im Arbeitsspeicher. Zum Schutz neu eingeschobener Disketten enthält CP/M deshalb eine Prüfroutine, die dies erkennt und den Schreibvorgang verhindert. Ein Warmstart sorgt dann für einen Austausch der Tabelle und ermöglicht das Beschreiben der Diskette.

Wenn allerdings von der neuen Diskette nur gelesen werden soll, ist ein Warmstart nicht erforderlich. Werden auf der neuen Diskette aber Schreibvorgänge durchgeführt, so muß vorher unbedingt ein Warmstart durchgeführt werden. Bei MP/M muß beachtet werden, daß keine Disketten eingeschoben oder ausgetauscht werden dürfen, ohne daß ein sogenanntes Rücksetzen der Disketten (disk reset - wird später beschrieben) durchgeführt wird.

CP/M Plus übernimmt diese Funktion, so daß bei einem Diskettenwechsel kein Warmstart erforderlich ist.

Nach einem Warmstart („^C") erscheint sofort wieder das Bereitschaftszeichen (A>). Nun können die Disketten in allen vorhandenen Laufwerken beschrieben werden. Hierbei muß beachtet werden, daß durch ^C ein eventuell vorher eingegebenes ^P rückgängig gemacht wird.

Durch die Eingabe von „B:" kann auf Laufwerk B umgeschaltet werden:

```
A>B:↵
B>
```

Nun ist das Laufwerk B zugeschaltet. Durch die Eingabe des DIR-Befehls kann überprüft werden, welche Dateien sich auf der Diskette in Laufwerk B befinden:

```
B>DIR↵
NOT FOUND
B>
```

Die Systemmeldung „NOT FOUND" sagt aus, daß mit dem DIR-Befehl keine Datei auf der Diskette gefunden wurde. In Laufwerk B befindet

sich dann eine leere Diskette. CP/M setzt (sofern nicht anders beschrieben) immer voraus, daß sich die Datei auf einer Diskette in dem jeweils zugeschalteten (aktuellen) Laufwerk befindet. Wenn also eine Datei gesucht wird und sie sich auf der Diskette in Laufwerk A befindet, kann entweder vorher Laufwerk A zugeschaltet oder die Laufwerkbezeichnung vor die Dateibezeichnung gesetzt werden. Soll beispielsweise das Programm PIP.COM in Laufwerk A gesucht werden, so lautet der entsprechende DIR-Befehl (durch ein Leerzeichen zwischen DIR und Dateibezeichnung getrennt):

```
B>DIR A:PIP.COM↵
A:PIP COM
B>
```

Genauso kann das Programm gefunden werden, wenn vor der Eingabe des DIR-Befehls das Laufwerk A zugeschaltet wird:

```
B>A:↵
A>DIR PIP.COM↵
A:PIP COM

A>
```

Wenn auf ein Laufwerk umgeschaltet wird, das keine Diskette enthält, dann wartet das System so lange (Kontrollampen des Laufwerks leuchten auf, und die Tastatur ist abgeschaltet), bis ein Kaltstart eingeleitet wird (vgl. Anfang des Kapitels).

Steht nur ein Laufwerk zur Verfügung, so besteht keine Wahl – die Systemdiskette muß herausgenommen und eine neue Diskette eingeschoben werden (vgl. auch Kapitel 2 für nähere Einzelheiten). Mit CP/M können bei Verfügbarkeit nur eines Laufwerks keine Kopien direkt von Disketten hergestellt werden. Spezielle Programme ermöglichen jedoch das Kopieren einer gesamten Diskette.

KOPIEREN

Erstellen einer Dateikopie

Es besteht oft die Notwendigkeit, Kopien von Disketten zu erstellen. So sollte auch nach der Erstellung einer Datei oder eines Programms eine

Kopie angelegt werden, um diese separat aufzubewahren (für den Fall, daß das Original verloren geht oder versehentlich zerstört wird).

Der korrekte Ablauf könnte wie folgt aussehen:

- Nach Erhalt eines neuen Programms oder einer neuen Datei sollte man sofort eine Kopie erstellen. Es empfiehlt sich, nur mit der Kopie zu arbeiten und das Original gut zu verwahren.
- Nach dem Erstellen oder Verändern eines Programms sollte man ebenfalls, bevor man das System verläßt, eine Kopie erstellen, die – sorgfältig mit Namen und Datum gekennzeichnet – verwahrt werden sollte.

Der PIP-Befehl wird für Kopiervorgänge eingesetzt. PIP ist die Abkürzung von „Peripheral Interchange Program". Das Programm wird durch die Eingabe von „PIP" gestartet. Damit das PIP-Programm ablaufen kann, muß es sich bereits auf der Diskette mit der Dateibezeichnung PIP.COM befinden (im allgemeinen auf der Systemdiskette).

In früheren Abschnitten dieses Kapitels wurde eine Datei mit der Bezeichnung BEISPIEL.TXT auf Laufwerk B generiert. Hiervon soll nun eine Kopie erstellt und auf der Diskette in Laufwerk A gespeichert werden. Der Kopiervorgang wird mit Hilfe des PIP-Befehls ausgeführt. Da sich das Programm PIP.COM auf der Diskette in Laufwerk A befindet, muß beim Standard-CP/M zuerst auf Laufwerk A umgeschaltet werden:

```
B>A:↲
A>
```

Nun kann PIP gestartet werden:

```
A>PIP↲
*
```

Der Stern (*) ist das Bereitschaftszeichen des PIP-Programms und zeigt, daß PIP geladen ist. Es kann jetzt eine PIP-Anweisung eingegeben werden, die den Kopiervorgang auslöst. Einfache PIP-Anweisungen haben die Form:

d:Kopiename = d:Originalname

Kopiename und Originalname sind Argumente und stehen für aktuelle Dateibezeichnungen. Die beiden vorherstehenden Zeichen d stehen für die anzusprechenden Laufwerke (drive specification). PIP kopiert immer von einer Originaldatei zu einer Kopierdatei. Die Originaldatei muß bestehen, während die Kopierdatei mit dem gewünschten Namen von PIP angelegt wird.

```
*A:KOPIE.TXT = B:BEISPIEL.TXT↵
```

Diese Anweisung teilt dem PIP-Programm mit, daß eine Kopie von der Datei mit der Dateibezeichnung „BEISPIEL.TXT“ von Laufwerk B auf Laufwerk A mit der Bezeichnung „KOPIE.TXT“ erstellt werden soll. Wenn bereits auf Laufwerk A geschaltet ist, so kann der Befehl in verkürzter Form eingegeben werden:

```
*KOPIE.TXT = B:BEISPIEL.TXT↵
```

PIP setzt voraus, daß die Kopierdatei auf dem gerade zugeschalteten Laufwerk erstellt werden soll (current drive = aktuelles Laufwerk). Da sich die Datei „BEISPIEL.TXT“ jedoch auf der Diskette in Laufwerk B befindet, muß dies angegeben werden. Es empfiehlt sich, die Laufwerkkennzeichnungen immer vollständig anzugeben. Wenn das PIP-Programm den Kopiervorgang beendet hat, wobei der Inhalt der Datei BEISPIEL.TXT in die Datei KOPIE.TXT übertragen wurde, meldet es sich wieder mit dem Bereitschaftszeichen *. PIP ist nun für weitere Anweisungen bereit. Durch das Drücken der RETURN-Taste (bzw. CR) kann das PIP-Programm wieder verlassen werden:

```
*↵
A>
```

Unabhängig davon, welche Laufwerke beim Kopieren angesprochen wurden, springt das System wieder zu dem Laufwerk zurück, das vor Aufruf des PIP-Programms zugeschaltet war, in diesem Fall also Laufwerk A.

Soll nur eine PIP-Operation durchgeführt werden, kann ein abgekürzter Befehl eingegeben werden, wobei sowohl das Wort PIP als auch die eigentliche PIP-Anweisung in einer Zeile geschrieben werden können:

```
A>PIP A:KOPIE.TXT = B:BEISPIEL.TXT↵
A>
```

Nach dem Kopiervorgang erscheint sofort wieder das Bereitschaftszeichen von CP/M.

Bei jedem Zugriff auf ein Laufwerk leuchtet die entsprechende Kontrolllampe auf und erlischt danach wieder. Dieser Vorgang wiederholt sich wechselseitig. Das PIP-Programm kopiert die gewünschten Dateien in einzelnen Segmenten, sogenannten Blöcken (blocks). Wenn die zu kopierende Datei umfangreich ist, muß PIP mehrmals auf die entsprechenden Laufwerke zugreifen, um jeweils einen Block zu kopieren. Nun wurde eine Kopie der Datei BEISPIEL.TXT mit der Dateibezeichnung KOPIE.TXT erstellt, und es soll eine Kopie der Datei KOPIE.TXT auf Laufwerk B erstellt werden:

```
A>PIP B: = A:KOPIE.TXT↵
A>
```

Die Spezifizierung einer Bezeichnung für die Kopie ist hier nicht notwendig, weil die Kopie die gleiche Bezeichnung wie das Original erhalten soll. Dieser Befehlstyp wird im allgemeinen häufiger angewandt als jeder andere.

Bei vollständiger Angabe von Ziel- und Quellendatei würde der Befehl für das obige Beispiel folgendermaßen lauten:

```
A>PIP B:KOPIE.TXT = A:KOPIE.TXT↵
A>
```

Beide Befehlstypen generieren die neue Datei KOPIE.TXT auf der Diskette in Laufwerk B, die eine vollständige Kopie der Datei KOPIE.TXT auf der Diskette in Laufwerk A ist. In den Fällen, in denen die Originalbezeichnung erhalten bleiben soll, ist die Angabe der Dateibezeichnung für die Kopie überflüssig. Die Laufwerksbezeichnungen jedoch müssen in jedem Fall angegeben werden und in diesem Fall unterschiedlich sein (es dürfen auf einer Diskette keine Dateien mit identischer Dateibezeichnung stehen).

Wird ein Kopiervorgang ohne Angabe einer neuen Dateibezeichnung

oder einer Laufwerksspezifizierung gestartet, so wird die Fehlermeldung „INVALID FORMAT“ mit einem entsprechenden Fehlercode ausgegeben. Eine solche Fehlermeldung kann durch die Betätigung von RUBOUT (oder DELETE) quittiert werden.

Die häufigsten Kopiervorgänge werden als vollständige Kopiervorgänge von Diskette zu Diskette (oder von Platte zu Diskette und umgekehrt) durchgeführt, weil hiermit sehr schnelle Kopien (Back-Up-Versionen) von Dateien erstellt werden können. Genauso wichtig wie der ERAse-Befehl (Befehl zum Löschen einzelner oder mehrerer Dateien) ist deshalb der Befehl zum Kopieren ganzer Disketten (z.B. wenn bei irrtümlicher Fehlanwendung des ERAse-Befehls ganze Dateien oder die gesamte Diskette gelöscht wird; weitere Hinweise in Kapitel 7).

Kopieren vollständiger Disketteninhalte

Wie bereits erläutert, kann z.B. die Datei KOPIE.TXT von Laufwerk A nach Laufwerk B mit dem folgenden Befehl kopiert werden:

```
A>PIP B: = A:KOPIE.TXT↵
A>
```

Damit wurde eine „backup“-Kopie der Datei KOPIE.TXT erstellt.

Zum Kopieren ganzer Disketteninhalte wird eine Kopieranweisung benötigt, die so allgemein ist, daß sie sich auf mehrere Dateien bezieht (filename match). Eine allgemeine Dateibezeichnung, die sich z.B. auf alle Dateien der angesprochenen Diskette bezieht, ist „*.*“. Die Kopieranweisung lautet dann:

```
A>PIP B: = A:*.*↵
```

Das Symbol * (wildcard, Joker) schließt dabei jede Bezeichnung dieses Feldes ein. Mit diesem Befehl werden alle Dateien der Diskette von Laufwerk A mit den gleichen Bezeichnungen auf die Diskette von Laufwerk B kopiert. (Es gelten in diesen Fall jedoch zwei Einschränkungen: Es werden nur Dateien ohne System-Attribut und vom augenblicklichen Benutzerbereich kopiert.) Soll der gesamte Inhalt einer Diskette auf einem anderen Laufwerk kopiert werden, dann sollte darauf geachtet werden, daß die dort eingelegte Diskette noch keine Dateien enthält (also „leer“ ist). Wird dies nicht beachtet, ergeben sich folgende Probleme:

1. Enthält B bereits Dateien, dann muß gewährleistet sein, daß auf B noch genügend Speicherplatz vorhanden ist. Eine Standarddiskette kann je nach Typ des betreffenden Laufwerks zwischen 160 KByte und 1,2 MByte aufnehmen (ausgenommen der Directory). Wird der STAT-Befehl ausgeführt, so wird der noch verfügbare Speicherplatz angezeigt.
2. Enthält B bereits eine Datei mit einer Dateibezeichnung, die identisch ist mit der einer Datei, die von A hinzukopiert werden soll, so wird der Kopiervorgang vom System abgebrochen.

Ist das System auf Laufwerk A geschaltet (wie im obigen Beispiel), so kann der Befehl nochmals abgekürzt werden:

```
A>PIP B: = *.*↵
```

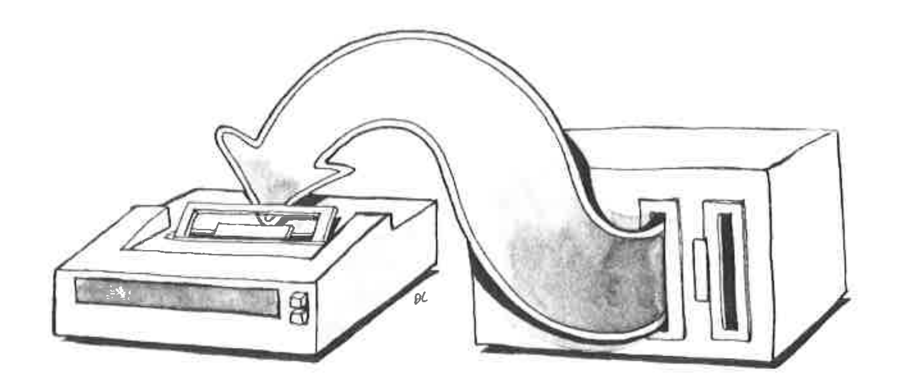
Beide Befehle suchen alle Dateien auf der Diskette in Laufwerk A und generieren hiervon auf Laufwerk B Kopien mit gleichlautenden Dateibezeichnungen.

Existiert bereits eine Datei auf der Diskette in Laufwerk B mit dem gleichen Namen, so wird diese gelöscht.

Dateibezeichnungen bestehen aus maximal 8 Zeichen vor und maximal 3 Zeichen nach dem Punkt. Das erste Ersatzsymbol * korrespondiert mit allen 8 Zeichen vor dem Punkt, während das zweite Symbol * sich auf alle drei Zeichen nach dem Punkt bezieht. Dabei ist zu beachten, daß Leerzeichen auch als Zeichen erkannt werden.

Mit diesem Befehl werden also alle Dateien der Diskette kopiert. Das CP/M-System selbst ist keine Datei. Es befindet sich auf reservierten Spuren der Diskette. Soll das gesamte CP/M-System kopiert werden, dann muß ein spezielles Programm gestartet werden.

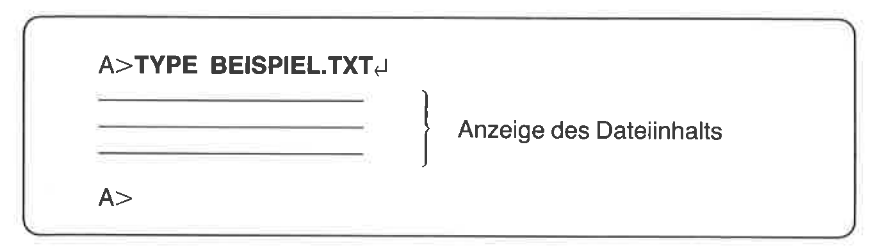




AUSDRUCKEN EINER DATEI

Bei der richtigen Anwendung der zuvor beschriebenen Befehle existieren jetzt zwei Kopien der Datei KOPIE.TXT (BEISPIEL.TXT und KOPIE.TXT auf Laufwerk B). Vor der Anwendung des ERAse-Befehls soll zunächst erläutert werden, wie ein Dateiinhalt über einen Drucker ausgegeben werden kann. Hierzu gibt es zwei Möglichkeiten.

Am einfachsten ist es, durch gleichzeitiges Drücken der CTRL- und der P-Taste (^P) den Drucker zuzuschalten. Alles, was von diesem Augenblick an auf dem Tastenfeld eingegeben wird und auf dem Bildschirm erscheint, wird mit dem Drucker gedruckt. Durch Eingabe des DIR-Befehls ist die korrekte Arbeitsweise des Druckers sehr einfach zu überprüfen. Mit dem TYPE-Befehl kann dann ein Dateiinhalt ausgedruckt werden:



Hierbei wird die Datei sowohl auf dem Bildschirm als auch auf dem Drukker ausgegeben.

Wird ^P nochmals gedrückt, wird der Drucker wieder abgeschaltet. Die Eingabe des TYPE-Befehls führt dann nur zu einer Ausgabe auf dem Bildschirm (allerdings mit viel höherer Geschwindigkeit). Diese Methode ist zwar sehr einfach, jedoch beim Ausdrucken mehrerer Dateien sehr umständlich. Hierzu kann eine Variante des PIP-Befehls eingesetzt werden:

```
A>PIP LST: = KOPIE.TXT↵
```

Dieser Befehl sendet den Inhalt der Datei KOPIE.TXT zum logischen „LIST-Gerät“ (listing device), dem Drucker. Falls dies nicht ausgeführt wird, muß der Drucker erst noch mit einer speziellen Routine zugewiesen werden (vgl. STAT-Befehl in Kapitel 2). Wird zusätzlich der Parameter P angefügt:

```
A>PIP LST: = Kopie.TXT[P]↵
```

so wird die Datei „seitenweise“ gedruckt (d.h. nach jeder Seite erfolgt ein Formularvorschub). Alle Dateien können zum Drucker gesendet werden, indem anstelle des Dateinamens die Symbole *.* eingegeben werden:

```
A>PIP LST: = B:*.*↵
```

LÖSCHEN VON DATEIEN

Angenommen, die Datei KOPIE.TXT befindet sich auf der Diskette von Laufwerk A, und auf der Diskette von Laufwerk B befinden sich die Dateien KOPIE.TXT und BEISPIEL.TXT, so daß eine der beiden Dateien gelöscht werden kann. Vorher sollte dies nochmals anhand der Inhaltsverzeichnisse überprüft werden:

```
A>DIR↵
A: KOPIE TXT
A>B:↵
B>DIR↵
B: BEISPIEL TXT : KOPIE TXT
B>
```

Zum Löschen der Datei KOPIE.TXT im Laufwerk B kann der folgende Befehl eingegeben werden:

```
B>ERA KOPIE.TXT↵
```

Besteht diese Datei nicht, so wird die Fehlermeldung „FILE NOT FOUND“ angezeigt.

Sollen Dateien auf einer anderen Diskette gelöscht werden, dann kann das Laufwerk der anzusprechenden Diskette mit angegeben werden:

```
B>ERA A:KOPIE.TXT↵
```

Dieser Befehl löscht die Datei KOPIE.TXT auf Laufwerk A. Alle Dateien einer Diskette können bei der Angabe der allgemeinen Dateibezeichnung gelöscht werden:

```
B>ERA *.*↵
```

Dieser Befehl löscht alle Dateien auf der Diskette von Laufwerk B, vorausgesetzt, sie sind nicht schreibgeschützt. (ERA löscht jedoch keine Dateien, die unter einer anderen Benutzernummer stehen.)

DAS CP/M-KONZEPT

Der interne Ablauf

Mit der Kenntnis über den Systemstart von CP/M (oder MP/M), das Generieren, Umbenennen und Löschen von Dateien, das Kopieren von Dateien und ganzer Disketteninhalte ist die innere Struktur des Ablaufs von CP/M noch weitgehend unbekannt.

CP/M reagiert auf eine Vielzahl von Instruktionen wie: Aufsuchen einer Datei auf einer Diskette über eine Dateizuordnung, Anzeige der Datei, Kopieren einer Datei usw.. Wie läuft jedoch eine einzelne Instruktion ab, z.B. beim Auflisten oder Kopieren einer Datei?

Wird z.B. die Befehlsfolge „TYPE BEISPIEL.TXT“ eingegeben, so wird eine Menge einzelner Instruktionen ausgeführt, die alle unter der Routine TYPE zusammengefaßt sind. Das Betriebssystem akzeptiert die eingegebene Zeichenfolge und liest diese nach dem Drücken der Return-Taste. Danach erkennt das Betriebssystem zunächst nur, daß der Befehl TYPE eingegeben wurde, und springt zur Routine mit den einzelnen TYPE-Instruktionen. Von dieser wird dann „BEISPIEL.TXT“ gelesen, und es wird ein Suchvorgang nach dieser Datei auf der Diskette des bisher zugeschalteten Laufwerks gestartet. Nachdem die Datei „BEISPIEL.TXT“ aufgefunden wurde, beginnt das Betriebssystem mit dem Lesen einzelner Dateiblöcke und sendet sie zum Bildschirm (console device). Der Dateiinhalt wird dann Block für Block übertragen, bis das

Ende der Datei erreicht ist. Das Bildschirmterminal gibt während der Übertragung bereits den Dateiinhalt aus. Wurde der Drucker zum Bildschirm parallel geschaltet (durch CTRL-P), dann wird alles zum Bildschirm gesendet und gleichzeitig auch auf dem Drucker ausgegeben.

CP/M ist ein komplexes Programm, das zum Teil sehr einfach aufgebaute Dienstprogramme (utility programs) aufruft und überwacht. Das System belegt einen bestimmten Bereich im Arbeitsspeicher innerhalb des Rechners, um so die gewünschten Programme schnell laden, starten und verwalten zu können.

Muß z.B. für das Kopieren einer Datei das PIP-Programm ausgeführt werden, so wird nach der Eingabe des PIP-Befehls das PIP-Programm in den internen Speicher geladen und dort gestartet. PIP wird auch als transienter Befehl bzw. transientes Programm bezeichnet. Es ist ein Programm in Maschinensprache, das genauso wie ein CP/M-Befehl ausgeführt wird. Der wesentliche Unterschied zu den residenten CP/M-Befehlen (z.B. DIR, ERA) besteht darin, daß es nicht Teil des residenten CP/M-Bereichs ist und auf der Diskette als Datei mit der Kennung „COM“ (PIP.COM) verfügbar sein muß. Weitere transiente Befehle sind z.B. STAT, SUBMIT oder SYSGEN.

Transiente Befehle

Transiente Befehle sind eigentlich Programme in Maschinensprache (Assemblersprache). Nach dem Assemblieren (Übersetzen in Maschinencode) und Testen wird das Maschinenprogramm (jetzt im Objektcode) mit dem LOAD-Befehl in den internen Arbeitsspeicher des Systems (auch Transient Program Area genannt) geladen. Es erhält den Typ „COM“. Dieses Programm kann auch über die Angabe des Dateinamens (ohne die Kennung „COM“) direkt geladen und ausgeführt werden. So ist z.B. das Programm PIP.COM als transienter Befehl direkt durch die Eingabe von „PIP“ ausführbar:

```
A>PIP↵
```

Es können so eigene beliebige Befehle bzw. Programme auf der Diskette untergebracht werden, z.B. ein Textprozessor oder ein Übersetzer für eine Programmiersprache (CBASIC, Microsoft-BASIC usw.). Das Textverarbeitungssystem Wordstar verfügt z.B. über einen transienten Befehl, nämlich WS (Programmname WS.COM). Es wird durch die Eingabe von

```
A>WS↵
```

gestartet.

Innerhalb des CP/M-Systems sind in Wordstar vielfältige Funktionsmöglichkeiten verfügbar. Ein anderes Beispiel ist das Microsoft-BASIC, das im CP/M-System als transienter Befehl MBASIC.COM aufgerufen werden kann. Für den Start und Ablauf wird dann nur eingegeben:

```
A>MBASIC↵
```

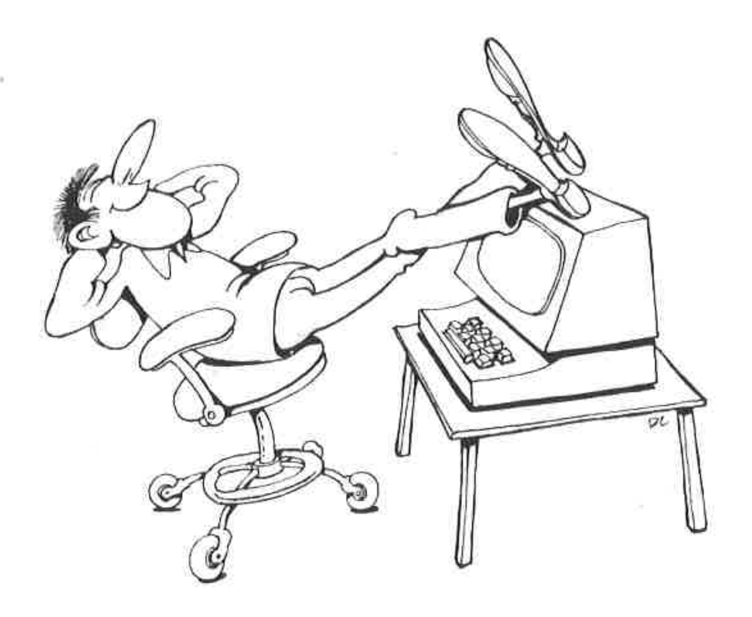
Auf jeden Fall sollten die jeweiligen Manuale (Softwarehandbücher) zu aktuellen Fragen bezüglich neuer Software herangezogen werden.

Abschalten des Systems

Soll das System abgeschaltet werden, dann müssen in jedem Fall vor dem Abschalten die Disketten aus allen Laufwerken herausgenommen werden. Wird dies nicht beachtet, so können die beim Abschaltvorgang auftretenden Spannungsspitzen wichtige Informationen auf der Diskette zerstören.

Sind alle Disketten herausgenommen, so können zunächst die Peripheriegeräte und dann der Computer gefahrlos abgeschaltet werden.

Wichtig: Vor dem Abschalten sollten noch von allen neu generierten Dateien Sicherungskopien (backup copies) erstellt werden.



BENUTZER-PRÜFLISTE

Die folgende Benutzer-Prüfliste faßt nochmals alle wesentlichen Vorsichtsmaßnahmen und wichtigen Abläufe zusammen. Sie sind unter allen Umständen zu beachten und sollten von jedem Benutzer befolgt werden:

Einschalten des Systems

Es muß gewährleistet sein, daß

- ☐ keine Disketten in den Laufwerken bei Zuschalten der Netzversorgung eingeschoben sind,
- ☐ eine fehlerfreie Systemdiskette verfügbar ist,
- ☐ eine oder mehrere leere Disketten verfügbar sind,
- ☐ alle Kabel ordnungsgemäß angebracht sind,
- ☐ alle Schalterpositionen an Drucker und Bildschirmterminal korrekt eingestellt sind.

Nutzung des Systems

Es ist sehr wichtig, daß

- ☐ die Kopien aller genutzten Disketten verfügbar sind,
- ☐ die Dateien während des Editierens ständig gesichert werden,
- ☐ die Disketten vollständig mit Titel, Datum und Inhalt beschriftet sind.

Nutzung neuer Programme

Wichtige Vorsichtsmaßnahmen:

- ☐ Erstellen einer Kopie vor der ersten Nutzung,
- ☐ Verwahren des Originals an einem sicheren Ort.

Abschalten des Systems

Es ist sehr wichtig, daß

- ☐ eine Sicherungskopie von allen neu generierten Dateien erstellt wurde,
- ☐ die Disketten aus allen Laufwerken herausgenommen worden sind.

ZUSAMMENFASSUNG

Nun sind die wichtigsten Arbeitsgänge beim Umgang mit einem CP/M-System erklärt worden: Ein- und Ausschalten des Systems, Benutzung

der einfachsten CP/M-Befehle und -Dienstprogramme wie DIR, ERA, PIP, ED und die Bedeutung spezieller Funktionen wie DELETE und CTRL-C (^C).

Darüber hinaus ist nun bekannt, wie Dateiinhalte über den Bildschirm oder Drucker aufgelistet und wie Kopien einzelner Dateien erstellt werden können.

Mit diesen Grundlagen können alle möglichen Anwenderprogramme ohne Schwierigkeiten gestartet, verarbeitet und abgeschlossen werden. Weitere Kenntnisse sind für den Anwender daher überflüssig. Derjenige jedoch, der einen tieferen Einblick in das Betriebssystem haben möchte, sollte sich mit den weiteren Möglichkeiten des Betriebssystems in den folgenden Kapiteln auseinandersetzen.



Kapitel 2

CP/M- und MP/M-Funktionen

EINFÜHRUNG

In diesem Kapitel werden wir die CP/M-Komandosprache kennenlernen mit den dafür benötigten Steuerzeichen und Befehlen. Dazu gehören Befehle zur Manipulation von Plattendateien wie ERA, REN, STAT und DIR, zur Arbeit auf Assembler-Ebene wie ASM, LOAD, DUMP, DDT und SAVE sowie zur automatischen Ablaufsteuerung (SUBMIT, XSUB). Steuerzeichen und Befehle werden in Tabellen zusammengefaßt.

Die Arbeitsweise eines jeden Befehls werden wir genau untersuchen. Dabei ist es für Sie nicht wichtig, diese im einzelnen zu behalten, sondern vielmehr, die Befehle sinnvoll einsetzen zu können.

Arbeiten Sie nur gelegentlich mit CP/M, so genügen folgende Grundkenntnisse:

- die fünf häufigsten Steuerzeichen (siehe folgenden Abschnitt)
- Löschen von Dateien mit ERA
- Umbenennen von Dateien mit REN
- Überprüfen des freien Dateispeicherraums mit STAT bzw. SHOW
- Kopieren eines CP/M-Systems

Für einen tieferen Einstieg ist es ratsam, dieses Kapitel ganz durchzuarbeiten. Als CP/M-Benutzer werden Sie anfangs noch häufiger Details nachlesen müssen. Eine Kurzform der Befehlsbeschreibungen finden Sie im Kapitel 6.

Bei der Beschreibung der Befehle gehen wir vom CP/M 2.2 für die 80er-Mikroprozessor-Familie aus. Diese Befehle gibt es, mit wenigen Ausnahmen, auch bei CP/M-Plus, CP/M-86, CCP/M und MP/M. Auf spezielle Erweiterungen werden wir später eingehen.

Als erstes werden wir die Befehlseingabe besprechen, dann eine Klassifizierung der Befehle vornehmen und schließlich auf den erweiterten Befehlsvorrat der CP/M-Familie eingehen.

BEFEHLE (Commands)

Befehle werden in diesem Buch der Übersicht halber alle in GROSS-BUCHSTABEN geschrieben. Befehlszusätze (arguments) werden hier klein geschrieben. Die in Schrägschrift gedruckten Argumente sind fakultativ (wahlweise). In Fällen, in denen mehrere Argumente in eckigen Klammern [] stehen, muß eines der Argumente angegeben werden.

Das Symbol ↵ steht für die Betätigung der Return- bzw. Enter-Taste, die einen Zeilenvorschub veranlaßt und meist eine Befehlseingabe abschließt und die Ausführung startet. Das Symbol ^ steht für die CTRL-Taste und zeigt an, daß diese Taste während der Eingabe des nachfolgenden Zeichens gedrückt sein muß. (^X bedeutet gleichzeitiges Drücken der Tasten CTRL und X.)

Bei der Befehlseingabe können die in der Tabelle Abb. 2.1 beschriebenen Steuerzeichen verwendet werden. Wenn z.B. der folgende PIP-Befehl eingegeben wurde:

```
A>PIP B:NEU.NCD=
```

jedoch DAT anstatt NCD eingegeben werden sollte, können sie durch viermaliges Betätigen der DEL-Taste die letzten vier Zeichen löschen. Das jeweils gelöschte Zeichen wird dabei ausgegeben:

```
A>PIP B:NEU.NCD==DCN
```

Der Befehl kann jetzt vervollständigt und mit RETURN abgeschlossen werden:

```
A>PIP B:NEU.NCD==DCNDAT = A:ALT.DAT↵
```

Um die Zeile in der korrigierten Form noch einmal richtig auf dem Terminal zu sehen, kann man vor dem RETURN mit ^R die Wiederholung der Eingabe veranlassen:

```
A>PIP B:NEU.NCD==DCNDAT = A:ALT.DAT^R
PIP B:NEU.DAT = A:ALT.DAT↵
```

Eine andere Möglichkeit zur Korrektur der Eingabe, die vor allem für Bildschirmterminals geeignet ist, ist die Benutzung der BS-Taste (Backspace, Rücktaste). Hierbei wird das gelöschte Zeichen nicht noch einmal ausgegeben, sondern das Zeichen wird auf dem Bildschirm gelöscht, und der Cursor geht um eine Position zurück.

Soll die gesamte Eingabezeile gelöscht werden, so kann das mit ^U oder, besonders geeignet für Bildschirmterminals, mit ^X geschehen.

Auslösende Taste	Operation
DELETE (RUBOUT)	Löschen des letzten Zeichens mit Echo
BACKSPACE (BS)	Löschen des letzten Zeichens mit Cursor zurück
CTRL-U	Löschen der gesamten Zeile, Ausgabe von # auf der nächsten Zeile
CTRL-X	Löschen der gesamten Zeile und Rücksetzen des Cursors auf Zeilenanfang
CTRL-R	Wiederholung der Eingabezeile auf dem Terminal
RETURN oder LINEFEED	Abschluß der Eingabezeile
CTRL-E	Eine neue Zeile auf dem Terminal wird begonnen, ohne daß ein Befehl ausgeführt wird
CTRL-C	Bei Betätigung am Zeilenanfang: Warmstart für CP/M
CTRL-D	nur CCP/M und MP/M: DETACH
CTRL-P	Ein- und Ausschalten des Druckerechos. Druckerecho bedeutet, daß alle zum Terminal gesandten Zeichen auch zum Drucker gesendet werden.
CTRL-S	Stoppt die Ausgabe von Texten auf das Terminal
CTRL-Q	Startet die Ausgabe auf das Terminal nach ^S

Abb. 2.1: Eingabe-Steuerzeichen

Residente und nicht-residente Befehle

Wie bereits beschrieben, meldet sich CP/M mit der Angabe des Arbeitslaufwerks und dem Zeichen >. Das System ist damit zur Befehlsannahme bereit. Die durch RETURN abgeschlossene Befehlszeile wird dann unmittelbar vom System interpretiert. Die Syntax der Befehle hat folgende Form:

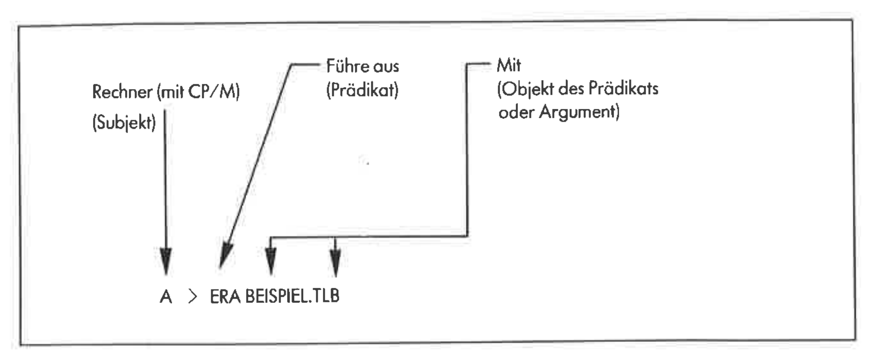


Abb. 2.2: „Rechner, lösche die Datei BEISPIEL.TLB"

CP/M unterscheidet zwischen residenten Befehlen (built-in commands), bei denen der zur Ausführung erforderliche Programmcode ständig im Speicher steht, und nicht-residenten Befehlen (transient commands), die vor der Ausführung von der Magnetplatte geladen werden. Die sechs residenten Befehle des CP/M sind:

TYPE
DIR
REN
ERA
SAVE (nur bei CP/M-80)
USER

MP/M und CCP/M kennen keine residenten Befehle. Bei der Systemgenerierung vom MP/M lassen sich jedoch einige Programme als sogenannte residente Systemprozesse einbinden.

Zum Lieferumfang der CP/M-Familie gehören u.a. folgende nicht-residente Befehle:

ED
PIP
STAT (nicht bei CP/M-Plus)
SHOW (nicht bei CP/M-80 und CP/M-86)
SUBMIT

Die Programme, die zur Abarbeitung der nicht-residenten Befehle benötigt werden, sind Dateien mit einer speziellen Kennung (siehe Abschnitt 2.2).

DATEINAMEN UND DATEIEN

Dateinamen

Bevor wir auf die einzelnen Befehle näher eingehen, hier ein paar Informationen zur Bezeichnungsweise von Dateien. Dateien (files) werden bei der CP/M-Familie nur auf Magnetspeicherplatten (Disketten oder Harddisks) verwaltet. Auf jeder Platte steht in einem dafür reservierten Bereich das Datei-Inhaltsverzeichnis (directory). Jeder Eintrag in der Directory beinhaltet den Dateinamen sowie die Information, wo auf der Platte die Datei abgespeichert ist. Der Zugriff auf einzelne Dateien erfolgt über den Namen. Der Name besteht aus 1 bis 8 Zeichen und einer optionalen Kennung von bis zu 3 Zeichen. Bei der Angabe des Dateinamens setzt man vor der Dateikennung einen Punkt. Die Tabelle Abb. 2.3 zeigt eine Anzahl von reservierten Dateikennungen. Dateinamen sollten nur aus großen Buchstaben bestehen. Da der Befehlsinterpreter die Eingabezeile intern in Großbuchstaben übersetzt, sind die folgenden zwei Befehle gleichbedeutend:

```
A>ERA DDT.COM↵

A>era ddt.com↵
```

Satzzeichen und Umlaute (deren Code im US-ASCII für andere Zeichen belegt ist) sind in Dateinamen nicht erlaubt.

Dateityp	Zuordnung	Beispiel
COM	Befehls-Datei für die 80er-Prozessoren	PIP.COM
CMD	Befehls-Datei für die 86er-Prozessoren	ED.CMD
68K	Befehls-Datei für 68000-Prozessor	DDT.68K
PRL	Befehls-Datei für MP/M-80	SDIR.PRL
ASM	Assembler-Quellprogramm für 80er-Prozessoren	EXAMP.ASM
A68	Assembler-Quellprogramm für 86er-Prozessoren	PATCH.A68
PRN	vom Assembler erzeugtes Listing	EXAMP.PRN
HEX	Programmcode in Hex-Format	EXAMP.HEX
H86	Programmcode in erweitertem Hex-Format	PATCH.H86
BAS	BASIC-Quellprogramm	GAME.BAS
INT	Zwischencode z.B. für Interpreter	GAME.INT
BAK	Sicherungsdatei (backup) des Editors	EXAMP.BAK
$$$	Zwischendatei nur während Programmlauf	LETTER.$$$

Abb. 2.3: Reservierte Dateikennungen mit Beispielen

Dateityp	Zuordnung	Beispiel
SUB	Textdatei für autom. Programmablauf (batch)	COMPIL.SUB
SPR	Systemfile für MP/M und CP/M-Plus	XDOS.SPR
RSP	„Residenter System Prozess“ (MP/M)	SPOOL.RSP
BRS	wie RSP, jedoch für Banking	SPOOL.BRS
REL	„Relocatible“ (verschiebbarer Objektcode)	BDOS3.REL
IRL	„Indexed Relocatible“ (spezieller REL-Code)	BASLIB.IRL

Abb. 2.3: Reservierte Dateikennungen mit Beispielen (Fortsetzung)

Bezeichnung von Dateigruppen

Einige Befehle lassen es zu, ganze Gruppen von Dateien gleichzeitig anzuwählen. Ein Zugriff auf solche Dateigruppen geschieht durch Einfügen von Fragezeichen (?) oder Sternen (*) bei der Angabe des Dateinamens. Die Zeichen werden auch mit dem Begriff „Wildcard“ (Joker) bezeichnet.

```
A>ERA O?T?.BAS↵
```

kann z.B. die Dateien OTTO.BAS, ORT.BAS und OSTE.BAS gleichzeitig löschen.

```
A>ERA ????????.???↵
```

löscht alle Dateien. Hierfür kann auch geschrieben werden:

```
A>ERA *.*↵
```

Der Stern wird hierbei vom Befehls-Interpreter automatisch in eine Reihe von Fragezeichen umgesetzt.

```
A>ERA Z*.*↵
```

löscht alle Dateien, die mit einem Z beginnen, und

```
A>ERA *.COM↵
```

löscht alle Dateien mit der Dateikennung „COM“. Zu beachten ist jedoch, daß alle Zeichen nach dem Stern (bis zum Punkt bzw. bei der Kennung bis zum Ende) ignoriert werden. So entspricht

```
A>ERA *ABC.*AC↵
```

dem Befehl

```
A>ERA *.*↵
```

Datei-Attribute

Jede Plattendatei kann (in der Directory) durch sogenannte Attribute gekennzeichnet werden. Die Kennzeichnung erfolgt normalerweise mit dem STAT-Befehl (bzw. SET bei CP/M-Plus). Um Dateien vor unbeabsichtigtem Löschen zu schützen, gibt es das Schreibschutz-Attribut (read only). Dateien mit dem System-Attribut werden bei der Ausgabe des Dateien-Inhaltsverzeichnisses nicht mit angelistet, was u.U. die Übersichtlichkeit erhöht. Auf die genaue Wirkung der Attribute werden wir später noch eingehen.

Benutzerbereiche

Die Dateiverwaltung der CP/M-Familie kann auf einem Laufwerk bis zu 16 verschiedene „Bereiche“ von Dateien verwalten, die z.B. verschiedenen Benutzern zugeordnet sind. Diese Bereiche sind von 0 bis 15 numeriert. Beim Kaltstart von CP/M wird automatisch der Benutzerbereich 0 angewählt. Mit dem USER-Befehl kann auf einen anderen Bereich umgeschaltet werden:

```
A>USER 3↵
```

Bei CP/M-Plus, CCP/M und MP/M wird der Benutzerbereich mit angezeigt:

```
3A>
```

Es ist im allgemeinen kein Zugriff auf Dateien eines anderen Benutzerbereichs möglich. Eine Ausnahme bildet der Bereich 0 (nicht bei CP/M-80). Dateien dieses Bereiches, bei denen das System-Attribut gesetzt ist, können auch unter einer anderen Benutzernummer gelesen werden.

CP/M-BEFEHLE

Der DIR-Befehl

Der DIR-Befehl dient zur Auflistung des Inhaltsverzeichnisses (Directory) eines bestimmten Plattenlaufwerks. Bei CP/M ist der DIR-Befehl resident, also immer verfügbar, während er beim MP/M und beim CCP/M als Befehlsdatei auf der Speicherplatte stehen muß. DIR listet nur die Dateien des augenblicklich angewählten Benutzerbereichs (siehe USER-Befehl), der bei CP/M nach dem Kaltstart automatisch 0 ist. Dateien mit gesetztem System-Attribut (siehe STAT-Befehl) werden nicht mit aufgelistet. Folgende Möglichkeiten der Eingabe gibt es:

```
A>DIR↵
```

listet alle Dateien vom Laufwerk A auf.

```
A>DIR B:↵
```

oder

```
A>B:↵
B>DIR↵
```

listet alle Dateien vom Laufwerk B auf. Mit

```
A>DIR *.DAT↵
```

werden alle Dateien mit dem Dateityp DAT von Laufwerk A angelistet. Für Laufwerk B läßt sich das gleiche so angeben:

```
A>DIR B:*.DAT↵
```

Die Ausgabe auf dem Bildschirm zeigt Abb. 2.4.

```
A>DIR↵
A:MOVCPM   COM:ASM       COM:DDT      COM:DUMP    COM
A:ED       COM:LOAD      COM:PIP      COM:STAT    COM
A:SUBMIT   COM:SYSGEN    COM:XSUB     COM:DISKDEF COM
A:DUMP     ASM:SINGLE    ASM:COPY     ASM:LIST    COM
A:FORMAT   COM:SAVEUSER  COM:USER     ASM:MEMR    COM
A:FILECOPY COM:SETDRIVE  COM:READ-ME  DOC:CONFIG  COM
```

Abb. 2.4: Ausgabe des Inhaltsverzeichnisses (directory) auf dem Bildschirm

Der TYPE-Befehl

Mit dem TYPE-Befehl kann eine Text-Datei auf dem Terminal ausgegeben werden. Der Text muß dazu in ASCII-Codierung (siehe Anhang) abgespeichert sein. Das Befehlsformat lautet:

```
TYPE d:Dateiname.Dateityp
```

„d:" gibt das Laufwerk (drive) an und kann weggelassen werden, wenn es sich um das vorgewählte Laufwerk handelt. Der TYPE-Befehl kann auch zum Auflisten des Textes auf dem Drucker dienen. Hierzu muß mit ^P das Druckerecho eingeschaltet werden, so daß Terminal und Drucker parallel arbeiten. Eine andere Möglichkeit zur Ausgabe auf dem Drucker bietet der PIP-Befehl.

Der REN-Befehl (rename)

Mit diesem Befehl können Dateinamen geändert werden. Das Format lautet:

```
REN neu = alt
```

wobei für „neu" der neue und für „alt" der bisherige Dateiname einzusetzen ist. Angenommen, auf dem Laufwerk A befindet sich die Datei DATEN1.TXT, die in DATEN2.TXT umbenannt werden soll. Der Befehl dazu lautet:

```
A>REN DATEN2.TXT = DATEN1.TXT↵
```

Eine Datei unter dem Namen DATEN1.TXT gibt es nach dieser Operation nicht mehr. Der Inhalt der Datei ist jedoch nicht geändert worden. Bei der Bezeichnung der Datei darf auch das Laufwerk mit angegeben werden, wobei es egal ist, ob bei „neu" oder bei „alt" oder bei beiden. Selbstverständlich darf die Laufwerksbezeichnung von „neu" und von „alt" nicht verschieden sein. Der REN-Befehl bricht mit Fehlermeldung ab, wenn

1. das Laufwerk schreibgeschützt ist oder
2. die Datei schreibgeschützt ist oder
3. die Datei „neu" schon vorhanden ist.

Der ERA-Befehl (erase)

Der Befehl ERA wird zum Löschen von Dateien eingesetzt. Das Format lautet:

ERA d:Dateibezeichnung

wobei d ein wahlweise angegebenes Laufwerk darstellt. Der Befehl

```
A>ERA PROG.TXT↵
```

löscht die Datei PROG.TXT auf Laufwerk A. Mit dem ERA-Befehl können durch Benutzung von ? oder * auch mehrere Dateien gleichzeitig gelöscht werden. Sollen z.B. alle Dateien vom Dateityp .ASM gelöscht werden, so genügt die Eingabe

```
A>ERA *.ASM↵
```

Alle Dateien eines Benutzerbereichs, sofern sie nicht schreibgeschützt sind, können mit dem Befehl

```
A>ERA *.*↵
```

gelöscht werden. Das System fordert jedoch in diesem Fall eine besondere Bestätigung

```
All (Y/N)?
```

und löscht die Dateien nur, wenn mit Y geantwortet wurde.

Wichtig: Der ERA-Befehl sollte mit äußerster Vorsicht angewendet werden, um ein irrtümliches Löschen zu vermeiden.

Der ERA-Befehl bricht mit einer Fehlermeldung ab, wenn das Laufwerk oder die Datei (nur bei CP/M-80) schreibgeschützt ist.

Der STAT-Befehl (status)

Der STAT-Befehl kann für mehrere verschiedene Aufgaben eingesetzt werden. Das sind im einzelnen:

- Anzeige des Laufwerkstatus und des freien Plattenspeicherplatzes
- Anzeige aller Dateien eines Laufwerks mit Länge und Attributen
- Anzeige der Plattenlaufwerksparameter
- Anzeige der aktiven Benutzerbereiche eines Laufwerks
- Setzen des Laufwerks in den Schreibschutz-Status
- Setzen von Datei-Attributen
- Anzeige der Zuordnung von Peripherie-Geräten (nur CP/M-80)
- Zuordnung von Peripherie-Geräten (nur CP/M-80)

Von diesen Funktionen werden wir hier nur die wichtigsten behandeln. Eine Beschreibung aller Funktionen finden Sie in Kapitel 6.

Zur Anzeige des Laufwerkstatus und des freien Plattenspeicherplatzes wird der STAT-Befehl ohne weitere Parameter eingegeben:

```
A>STAT↵
A:R/W,SPACE:144K
A>
```

Der Laufwerkstatus ist R/W (read/write): das bedeutet, daß für das Laufwerk sowohl Lese- als auch Schreiboperationen durchgeführt werden können. Im Gegensatz dazu steht R/O (read only) bei einem schreibgeschützen Laufwerk. Dieser Status ist nicht mit dem physikalischen Schreibschutz eines Laufwerks (z.B. Schreibschutz-Kerbe der Diskette) identisch. Der freie Plattenspeicherplatz (space) wird in KByte (1 KByte sind 1024 Bytes mit je 8 Bit) angegeben. Platten, die beschrieben werden, sollten ab und zu auf genügend großen freien Speicherplatz überprüft werden.

Wird der STAT-Befehl ohne Angabe eines Laufwerks gegeben, so wird der Status aller angewählten Laufwerke (logged drives) aufgelistet. „Logged Drives“ sind die Laufwerke, die seit dem letzten Warmstart angesprochen wurden.

```
A>B:↵
B>STAT↵
A:R/O,SPACE:144K
B:R/W,SPACE:73K
B>
```

Der Schreibschutz-Status (R/O) kann u.a. dadurch eingeschaltet sein, daß eine Diskette ohne nachfolgenden Warmstart (bzw. DSKRESET-Befehl bei MP/M) gewechselt wurde. In diesem Falle ist die Angabe des freien Speicherplatzes nicht korrekt.

Ein Laufwerk kann auch mit folgendem Befehl einen logischen Schreibschutz bekommen:

```
STAT d:=RO   bzw. bei CP/M-80:   STAT d:=R/O
```

d steht hier für ein Laufwerk (A...P).

Wird dem STAT-Befehl eine Dateibezeichnung angefügt, so werden die wichtigsten Parameter dieser Datei gelistet. Anstelle einer einzelnen Dateibezeichnung kann hier eine Dateigruppe angegeben werden.

```
A>STAT B:*.COM↵

Recs  Bytes  Ext    Acc
  48     6k    1    R/O  B:ED.COM
  55    12K    1    R/O  (B:PIP.COM)
 140    18K    2    R/W  B:BASIC.COM
Bytes Remaining On B: 116k
```

Die Spalte „Recs“ zeigt an, wieviel 128-Byte-Sätze (bei Disketten mit einfacher Schreibdichte identisch mit der Anzahl der Sektoren) die jeweilige Datei belegt. Mit der Spalte „Bytes“ wird angezeigt, wieviel Platz die Datei insgesamt auf der Diskette belegt. Die Angabe ist in KByte (= 1024 Bytes) gemacht und ergibt sich aus der Aufrundung der Dateilänge zum nächsten Blockende. Diese Blöcke sind CP/M-interne Einheiten und von der Laufwerkskapazität abhängig. Die Spalte „Ext“ gibt die Anzahl der

im Inhaltsverzeichnis (directory) gespeicherten Einträge für die entsprechende Datei an. Bei Laufwerken mit weniger als 256 KBytes Kapazität wird für jeweils 16 KBytes Dateilänge ein Directory-Eintrag benutzt. Die Gesamtanzahl der Directory-Einträge ist begrenzt. Die Spalte „Acc" (Access) gibt an, auf welche Art auf die Datei zugegriffen werden kann. Standard ist R/W, das heißt, die Datei kann nicht nur gelesen, sondern auch beschrieben oder gelöscht werden. Ist das Schreibschutz-Attribut gesetzt, so steht an dieser Stelle R/O. Es folgt der Dateiname selbst mit der entsprechenden Laufwerksangabe. Ist das System-Attribut gesetzt, so steht der Dateiname in runden Klammern. Als letzte Zeile wird der freie Speicherplatz auf der Diskette angegeben.

Mit dem folgenden Befehl wird eine Liste der wichtigsten Laufwerksparameter ausgegeben:

```
A>STAT DSK:↵
      A: Drive Characteristics
   1944: 128 Byte Record Capacity
    243: Kilobyte Drive Capacity
     64: 32 Byte Directory Entries
     64: Checked Directory Entries
    128: Records/ Extent
      8: Records/ Block
     26: Sectors/ Track
      2: Reserved Tracks
A>
```

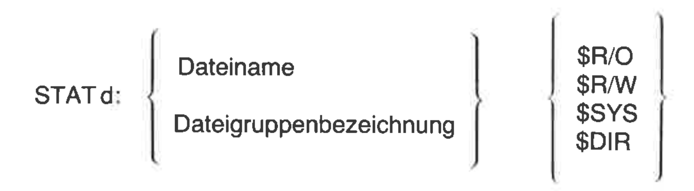
Record Capacity und Drive Capacity geben den insgesamt verfügbaren Speicherplatz in 128-Byte-Records bzw. in KBytes an. Die Anzahl der möglichen Directory-Einträge, von denen jeder 32 Bytes lang ist, steht in der 4. Zeile. Die Anzahl der zum Erkennen eines Diskettenwechsels zu überprüfenden Directory-Einträge steht in der 5. Zeile. Wie viele 128-Byte-Records ein Directory-Eintrag (auch Extent genannt) verwalten kann, steht in der 6. Zeile. Die sogenannten Blöcke (blocks) sind die kleinsten vom System vergebenen Platzeinheiten. Ihre Größe wird in Zeile 7 angegeben. Als nächstes folgt die Anzahl der 128-Byte-Records pro Spur auf dem Laufwerk. Die letzte Angabe sagt aus, wieviel Spuren, von der äußeren (Spur 0) aus gerechnet, für das System reserviert sind. Die Systemspuren werden vom Betriebsystem selbst nicht mit verwaltet.

STAT ermöglicht es, ohne Umschaltung der Benutzernummer (mit dem USER-Befehl) zu überprüfen, für welche Benutzer-Bereiche Dateien bestehen:

```
A>STAT USR:↵
Active User: 0
Active Files: 0 1 3
A>
```

Diese Angaben sagen aus, daß im Augenblick der Bereich 0 aktiv ist und daß für die Bereiche 0, 1 und 3 (des Laufwerks A) Dateien bestehen.

Der STAT-Befehl kann auch zum Setzen bzw. Rücksetzen der Datei-Attribute benutzt werden. Das allgemeine Format dazu ist:



Mit $R/O wird das Schreibschutz-Attribut gesetzt und mit $R/W zurückgesetzt. $SYS setzt das System-Attribut, und $DIR setzt es wieder zurück.

Eine Übersicht über die verfügbaren STAT-Funktionen bietet der STAT-Befehl selbst mit dem Befehl:

```
A>STAT VAL:↵
Temp R/O Disk: d:=R/O
Set Indicator  : d:filename.typ $R/O $R/W $SYS $DIR
Disk Status    : DSK: d:DSK:
User Status    : USR:
Iobyte Assign  :
CON: = TTY: CRT: BAT: UC1:
RDR : = TTY: PTR: UR1: UR2:
PUN : = TTY: PTP : UP1 : UP2:
LST  : = TTY: CRT: LPT : UL1:
A>
```

Diese Angaben sind bei den verschieden Versionen des Betriebssystems unterschiedlich. Die Zuordnung des sogenannten I/O-Byte (Iobyte Assign) gibt es nur beim CP/M-80; ihre Wirkungsweise ist zudem imple-

mentationsabhängig (z.T. auch gar nicht vorhanden), so daß wir hier nicht darauf eingehen werden.

Beim CP/M-Plus ist der STAT-Befehl durch SHOW und SET (zum Setzen von Attributen etc.) ersetzt worden. MP/M und CCP/M kennen sowohl STAT als auch SHOW und SET.

Der SUBMIT-Befehl

Der SUBMIT-Befehl erlaubt einen automatischen Ablauf von Befehlsfolgen. Mit einem Editor (ein Programm zur Erstellung von Text-Dateien, z.B. ED.COM) wird eine Text-Datei mit den abzuarbeitenden CP/M-Befehlen angelegt. Mit dem SUBMIT-Befehl kann jetzt die Abarbeitung gestartet werden, wobei die Zeilen aus der Text-Datei die Konsoleneingabe ersetzen. Betrachten wir diesen Ablauf anhand eines Beispiels. Die Text-Datei „COPY.SUB" beinhaltet folgende Befehle:

```
STAT  B:
PIP   B: = A:TELE.TXT
STAT  B:
DIR   B:
```

Der Aufruf und Ablauf dieser Befehlsfolge geschieht folgendermaßen:

```
A>SUBMIT COPY↵
A>STAT B:
B:R/W,SPACE:241K
A>PIP  B: = A:TELE.TXT
A>STAT  B:
B:R/W,SPACE:193K
A>DIR  B:
B:TELE     TXT
A>
```

In den Submit-Dateien können bis zu drei Variablen eingefügt werden. Sie werden mit $1, $2 und $3 bezeichnet. Für das obige Beispiel könnte die Text-Datei z.B. so aussehen:

```
STAT B:
PIP B:=A:$1
STAT B:
DIR B:
```

Der Aufruf

```
A>SUBMIT AUTO TELE.TXT↵
```

würde dann den gleichen Ablauf veranlassen.

Der PIP-Befehl

PIP (peripherial interchange program) ist ein universelles Programm zur Übertragung von Dateien und Ein/Ausgabezeichen. Eine genaue Beschreibung finden Sie in Kapitel 3.

Der ED-Befehl

Der Text-Editor ED wird detailliert in Kapitel 4 beschrieben.

ASM, LOAD, DUMP und DDT

Diese vier Programme sind gedacht für die Programmentwicklung in Assemblersprache. ASM übersetzt ein z.B. mit ED eingegebenes Programm in die Maschinensprache des jeweiligen Prozessors (bei CP/M-80 für den 8080) und erzeugt eine Datei, in der der Maschinencode hexadezimal codiert ist. Mit dem Befehl LOAD läßt sich daraus eine CP/M-Befehlsdatei herstellen. DUMP dient zur Auflistung von Objekt-Code-Dateien (z.B. Befehls-Dateien) in hexadezimaler Form. Mit dem Debugger („Entwanzer“, Testhilfe) DDT (dynamic debugging tool) lassen sich Programme testen und mögliche Fehler finden.

SYSTEM-GENERIERUNG

Beim Kaltstart, also dem ersten Anlaufen des Rechners nach dem Einschalten oder einem Hardware-Reset, muß das Betriebsystem von der Floppy oder Festplatte in den Arbeitsspeicher geladen werden. Beim CP/M-80 wird außerdem bei jedem Warmstart (z.B. nach ^C) ein Teil des Betriebsystems nachgeladen. Üblicherweise werden die äußeren Spuren der Magnetplatten für das System reserviert (Systemspuren). Geladen wird von Laufwerk A, d.h., auf den Systemspuren der Platte in Laufwerk A muß das Betriebssystem stehen. Da die Systemspuren nicht als CP/M-Datei angesprochen werden können, müssen spezielle Programme zur Generierung von Systemplatten vorhanden sein. Diese Programme sind abhängig vom Rechnertyp. Als Beispiel sei hier das CP/M-Programm SYSGEN zum Lesen und Schreiben der Systemspuren einer 8-Zoll-Single-Density-Diskette für CP/M-80 genannt. Der folgende Ablauf wird durch die Eingabe des SYSGEN-Befehls gestartet:

```
A>SYSGEN↵
SYSGEN VERSION 2.0
SOURCE DRIVE NAME (OR RETURN TO SKIP) A
   (Angabe des Quell-Laufwerks; wenn das System bereits im
   TPA-Bereich des Arbeitsspeichers steht, z.B. nach MOVCPM,
   kann auch RETURN eingegeben werden)
SOURCE ON A THEN TYPE RETURN
   (Nachdem die Quell-Diskette in A steckt, RETURN eingeben)
FUNCTION COMPLETE
   (Das System steht jetzt im TPA-Bereich des Arbeitsspeichers)
DESTINATION DRIVE NAME (OR RETURN TO REBOOT) B
   (Angabe des Ziel-Laufwerks)
DESTINATION ON B THEN TYPE RETURN↵
   (Nachdem die Zieldiskette in B steckt, RETURN eingeben)
FUNCTION COMPLETE
DESTINATION DRIVE NAME (OR RETURN TO REBOOT)↵
   (Es können jetzt noch weitere System-Disketten erzeugt
   werden. Mit RETURN wird das SYSGEN-Programm beendet.)
A>
```

Durch das Kopieren der Systemspuren mit SYSGEN werden die Directory und die gegebenenfalls vorhandenen Dateien nicht zerstört. Die Systemspuren können also auch nachträglich beschrieben bzw. geändert werden.

Bei CP/M-80 reicht in fast allen Fällen das Kopieren der Systemspuren zur Generierung von System-Platten aus. Bei den anderen Betriebssystemen der CP/M-Familie befindet sich auf den Systemspuren lediglich ein System-Lader; das eigentliche System ist eine Datei, die z.B. mit PIP kopiert werden muß. Bei CCP/M und MP/M, die ja keine residenten Befehle kennen, sollten die wichtigsten Befehlsdateien ebenfalls auf die neue Systemplatte kopiert werden.

MULTITASKING: MP/M UND CCP/M

CCP/M (Concurrent CP/M) und MP/M (Multi Programming System for Microcomputers) sind sogenannte Multitasking-Systeme, d.h. es können mehrere Prozesse (tasks) parallel (bzw. scheinbar parallel) ablaufen. MP/M ist zudem ein Multi-User-System (Mehrbenutzer-System). Bis zu 8 Benutzer können gleichzeitig (über eigene Terminals) an einem System arbeiten. Prozesse können ganze Programme sein oder auch Teile von Programmen mit bestimmten Aufgaben. Die Technik des (scheinbar) parallelen Prozeßablaufs wird auch als Timesharing bezeichnet: Der Prozessor verteilt seine Arbeitszeit auf mehrere Prozesse.

Die Arbeitsweise von CCP/M entspricht der eines MP/M-Systems mit nur einem Terminal (Benutzer). Alle Programme (von ganz wenigen Ausnahmen abgesehen), die auf CP/M laufen, sind auch auf MP/M und auf CCP/M lauffähig, vorausgesetzt das System benutzt einen verträglichen Prozessortyp.

Sowohl CCP/M als auch MP/M ermöglichen es, von einer Konsole aus mehrere Programme zu starten. Da nur jeweils ein Programm mit der Konsole kommunizieren kann, muß es nach dem Starten in den Hintergrund geschaltet werden (detached), so daß das Terminal wieder frei ist. Bei CCP/M kann ein solches Hintergrundprogramm Ausgaben auf ein „virtuelles Terminal" machen. Dieses virtuelle Terminal ist nicht real; die Terminal-Ausgabe wird lediglich in einem File zwischengespeichert und erst später auf das reale Terminal ausgegeben, sobald das Programm wieder im Vordergrund läuft (attached). Will ein Hintergrundprogramm unter MP/M Ausgaben auf das Terminal machen, so muß es warten, bis es wieder im Vordergrund läuft.

Das Detach-Steuerzeichen (CTRL-D)

Bei MP/M gibt es, im Gegensatz zu CCP/M, keine virtuellen Terminals im Hintergrund. Trotzdem lassen sich von einem Benutzer mehrere Programme starten. Der Vorgang dazu ist folgender:

Sobald ein Programm aufgerufen wurde, wird es durch Eingabe von CTRL-D in den Hintergrund geschaltet, das heißt „detached". Es meldet sich daraufhin wieder der „Terminal Process" (TMP) mit der Bereitschaftsmeldung. Es kann jetzt ein weiteres Programm aufgerufen und dann ebenfalls „detached" werden. (Bei MP/M-80 ist die max. Anzahl der parallel laufenden Programme abhängig von der Anzahl der Speicherbänke.) Die in den Hintergrund geschalteten Programme arbeiten natürlich nur so lange weiter, bis sie eine Konsoleein- oder -ausgabe machen wollen. Zurück in den Vordergrund (d.h. auf die Konsole) geschaltet werden sie entweder mit dem Befehl

```
0A>ATTACH programmname↵
```

wobei „programmname" für das Programm steht, das wieder in den Vordergrund geschaltet werden soll, oder durch die Eingabe von CTRL-D, während der TMP auf eine Eingabe wartet. In diesem Fall werden die Programme in der gleichen Reihenfolge, in der sie „detached" wurden, wieder „attached".

Der DSKRESET-Befehl

Wird eine Diskette gewechselt, so muß das Betriebsystem darüber informiert werden, daß die alten Laufwerksparameter nicht mehr stimmen.

Der Warmstart beim CP/M, z.B. durch CTRL-C ausgelöst, bedeutet stets ein Rücksetzen des Betriebssystems in den Anfangszustand und muß deshalb zum Diskettenwechsel benutzt werden. Bei CCP/M und MP/M gibt es ein Rücksetzen des ganzen Systems nicht mehr: Hier dient der Befehl DSKRESET dazu, bereits aktivierte (logged) Laufwerke wieder zu deaktivieren. Beim nächsten Zugriff auf ein nicht aktiviertes Laufwerk wird es automatisch aktiviert und seine Parameter festgestellt. Drives können nur dann ausgeloggt werden, wenn kein anderer Prozeß auf ihnen arbeitet. Die Syntax ist folgende:

```
DSKRESET
DSKRESET d:
DSKRESET d:,d:,d:...
```

Der erste Befehl setzt alle Laufwerke zurück. Der zweite Befehl setzt nur Laufwerk d (A...P) zurück. Mit dem dritten Befehl läßt sich eine Auswahl mehrerer Laufwerke zurücksetzen. Ist ein Laufwerk nicht frei, d.h. hat ein Prozeß noch offene Dateien auf diesem Laufwerk, so wird der Befehl mit einer Fehlermeldung abgebrochen. Beispiel:

```
0A>DSKRESET B:↲
Disk reset denied, Drive B: Console 2 Program PIP
```

Das bedeutet, daß kein Rücksetzen durchgeführt wurde, da das Programm PIP auf Konsole 2 offene Dateien auf Laufwerk B hat.

Im Gegensatz zum CP/M, bei dem der Warmstart nach dem Diskettenwechsel gegeben wird, muß DSKRESET vorher gegeben werden.

Der ABORT-Befehl

Mitunter kommt es vor, daß ein Programm vor seiner eigentlichen Beendigung abgebrochen werden muß. Wartet das Programm gerade auf eine Eingabezeile, so kann es mit CTRL-C beendet werden. Bei CP/M löst das einen Warmstart aus; bei MP/M und CCP/M erfolgt die Nachricht:

```
DDT: Abort (Y/N)?
```

Wird jetzt mit Y geantwortet, so endet das Programm. Wartet das Programm nicht auf eine Eingabe, so kann es von einer anderen Konsole aus mit dem ABORT-Befehl abgebrochen werden. Arbeitet das Programm im Hintergrund der eigenen Konsole, so kann der ABORT-Befehl auch hier gegeben werden. Die Syntax lautet:

```
ABORT programmname
ABORT programmname n
```

wobei „programmname“ das abzubrechende Programm ist und n die Konsolen-Nummer, die weggelassen wird, wenn das Programm im Hintergrund der eigenen Konsole läuft.

Echtzeituhr

CP/M-Plus, MP/M und CCP/M sind zum Betrieb einer Echtzeituhr ausgerüstet. Bei CP/M-Plus muß eine Hardware-Uhr vorhanden sein; bei MP/M und CCP/M wird vom System eine Uhr nachgebildet (Software-Uhr), die ggf. beim Kaltstart von einer Hardware-Uhr gestellt wird. Die Uhr stellt das Datum sowie die Uhrzeit bis zur Sekunde zur Verfügung. Zum Lesen und Stellen der Echtzeituhr dient bei MP/M und CCP/M der Befehl TOD (time of date) und bei CP/M-Plus der Befehl DATE. Die Syntax ist folgende:

```
        TOD                       Anzeige von Datum und Uhrzeit
        TOD P                     dauernde Anzeige der Uhr
        TOD MM/DD/YY HH:MM:SS     Stellen von Datum und Uhrzeit
bzw.
        DATE                      Anzeige von Datum und Uhrzeit
        DATE C                    dauernde Anzeige der Uhr
        DATE MM/DD/YY HH:MM:SS    Stellen von Datum und Uhrzeit
        DATE SET                  Interaktives Stellen der Uhr
```

Das Format der Anzeige und zum Stellen der Uhr ist

```
MONAT/TAG/JAHR STUNDE:MINUTE:SEKUNDE
```

jeweils in 2 Ziffern. Beispiel:

```
0A>TOD↵
Fri 01/12/84  14:22:09
```

Das Stellen der Uhr unter CP/M-Plus kann so aussehen:

```
A>DATE 02/09/83 10:30:00↵
Press any key to set time
Wed 02/09/83  10:30:00
```

Paßwort- und Zeiteinträge in der Directory

Die Betriebssysteme CP/M-Plus, MP/M und CCP/M haben die Möglichkeit, in den Datei-Inhaltsverzeichnissen (Directories) auf den Magnet-

platten zusätzliche Informationen, den Zugriff zu einzelnen Dateien betreffend, unterzubringen. Das können Geheimwörter (passwords) zum Schutz von Dateien vor unautorisierten Personen und Einträge über Erstellungs-, Zugriffs- und Änderungsdatum/-zeit von Dateien (time and date stamps) sein.

Die Steuerung dieser Einträge erfolgt mit dem SET-Befehl. Eine genaue Beschreibung dieses Befehls finden Sie in Kapitel 6. Hier soll jedoch an einem Beispiel die Einrichtung von Zeit-Einträgen auf Laufwerk B gezeigt werden.

Als vorbereitende Maßnahme muß bei CP/M-Plus das Programm INITDIR aufgerufen werden:

```
A>INITDIR B:↵
INITDIR WILL ACTIVATE TIME STAMPS FOR SPECIFIED DRIVE.
Do you want to re-format the directory on drive: B (Y/N)? Y↵
```

Alle weiteren Schritte gelten auch für MP/M und CCP/M. Die Platte muß als erstes ein sogenanntes Label bekommen: einen Namen nach eigener Wahl, der in der Directory gespeichert wird. Wir wählen den Namen „WORDSTAR“, weil auf der Platte das WordStar-Textverarbeitungsprogramm stehen soll.

```
A>SET B:[NAME=WORDSTAR]↵
Label for drive B:
Directory      Passwds  Stamp    Stamp    Stamp
Label          Reqd     Create   Access   Update
-------------- -------- -------- -------- --------
B:WORDSTAR.    off      on       off      on
```

Sollen in Zukunft alle Dateierstellungen und die jeweils letzten Änderungen als Zeiteinträge festgehalten werden, so wird die Platte durch folgenden Befehl dafür vorbereitet:

```
A>SET B:[CREATE=ON,UPDATE=ON]↵
Label for drive B:
Directory      Passwds  Stamp    Stamp    Stamp
Label          Reqd     Create   Access   Update
-------------- -------- -------- -------- --------
B:WORDSTAR.    off      on       off      on
```

Zum Schluß soll noch einmal darauf hingewiesen werden, daß die Directory-Zeiteinträge nur dann sinnvoll sind, wenn das System über eine Echtzeituhr verfügt. In vielen Systemen wird jedoch die Uhr nur per Software nachgebildet und muß dann nach jedem Kaltstart gestellt werden (siehe TOD- bzw. DATE-Befehl).

ZUSAMMENFASSUNG

In diesem Kapitel wurden zunächst alle von CP/M unterstützten Funktionen, von den Steuerzeichen bis zu den residenten Befehlen, behandelt. Neben allen Steuerzeichen sind vier der fünf residenten Befehle, nämlich DIR, TYPE, REN und ERA, sowie vier der wichtigsten nicht residenten Befehle, nämlich PIP, ED, SYSGEN und STAT bekannt. Die anderen fünf nicht residenten Befehle (ASM, LOAD, DUMP, DDT und SAVE) sind nur dann von Bedeutung, wenn Assemblerprogramme generiert und ausgeführt oder wenn die Möglichkeit der Steuerung über Befehlsdateien mit SUBMIT genutzt werden sollen. Eine vollständige Zusammenfassung dieser Befehle wird in Kapitel 6 gegeben.

Kapitel 3

Handhabung von Dateien

EINFÜHRUNG

In Kapitel 1 wurde bereits auf den PIP-Befehl zum Kopieren von Dateien eingegangen. Die wichtigste Anwendung von PIP ist das Kopieren oder genauer das Übertragen von Dateien.

Darüber hinaus bietet der PIP-Befehl jedoch noch viele andere Funktionsmöglichkeiten.

In diesem Kapitel soll auf alle Möglichkeiten des PIP-Befehls eingegangen werden. Auch wenn zunächst nicht alle Funktionen des PIP genutzt werden können, empfiehlt sich ein vollständiges Durcharbeiten dieses Kapitels, da die Kenntnis aller Möglichkeiten des PIP sehr wichtig ist. Auf diese Weise wird das Konzept von PIP weitgehend erfaßt. Danach können die interessanten Befehle nochmals im Detail studiert und getestet werden.

Mit PIP wird ein weiterer Bereich zur Dateibehandlung abgedeckt:

- das Aneinanderfügen einzelner Dateien (concatenation)
- das formatierte Ausdrucken von Texten (Auswertung von Tabs)
- das Abschneiden zu langer Druckzeilen für die Bildschirmausgabe (D-Option)
- das Ausdrucken mehrerer Dateien mit einem Befehl (PRN-Option)
- das Einfügen von Text in formatierte Seiten (P-Option)
- das Ausdrucken bis zu bestimmten Zeichenfolgen (Q-Option)

Grundlagen von PIP

PIP kann zunächst als Dateiübertragungsprogramm verstanden werden. PIP ist die Abkürzung für Peripheral Interchange Program. Wie bereits in der Programmbezeichnung angegeben, erlaubt PIP die Übertragung von Dateien zwischen zwei Geräteeinheiten. Bisher wurden für den PIP-Befehl nur Übertragungen von der Disketteneinheit behandelt, ohne auf die weiteren Möglichkeiten einzugehen.

Zunächst soll die wichtigste PIP-Funktion besprochen werden: das Kopieren von Dateien. Im Anschluß daran werden die PIP-Funktionen für die Übertragung von Dateien zwischen unterschiedlichen Geräteeinheiten analysiert.

KOPIEREN VON DATEIEN

Kopieren einer einzelnen Datei

PIP kann auf zwei Arten gestartet werden:

1. als einzelne Befehlszeile
2. als Programm

Im folgenden Beispiel wird PIP als Einzelbefehl eingesetzt:

```
A>PIP B:KOPIE1.BAK = DATEI1.TXT↵
A>
```

Hierbei wird vorausgesetzt, daß die Datei DATEI1.TXT über das selektierte Laufwerk A zugreifbar ist. Dieser Befehl teilt der PIP-Routine mit, daß eine Kopie der DATEI1.TXT erstellt und als Datei KOPIE1.BAK auf der Diskette von Laufwerk B generiert werden soll. Danach meldet sich CP/M wieder mit dem Bereitschaftszeichen (A>). Mit dieser Methode können schnell einzelne Dateien kopiert werden.

Genauso kann PIP als Programm ablaufen, so daß hierin dann die einzelnen Kopierbefehle als Folge eingegeben werden können:

```
A>PIP↵
*B:KOPIE2.BAK = DATEI2.TXT↵
*A: = B:BEISPIEL.BAS↵
*A: = B:PROG.FOR↵
*↵
A>
```

Nach der Eingabe von PIP↵ meldet sich dieses Programm mit dem Bereitschaftszeichen *. Danach können die einzelnen PIP-Befehle eingegeben werden. Mit der ersten Zeile wird beispielsweise die Erstellung einer Kopie KOPIE2.BAK auf Laufwerk B von der Datei DATEI2.TXT auf Laufwerk A veranlaßt. Mit der nächsten Zeile wird eine Kopie von der Datei BEISPIEL.BAS von Laufwerk B erstellt und mit der gleichen Dateibezeichnung auf der Diskette in Laufwerk A gespeichert. Die dritte Zeile erzeugt den gleichen Vorgang für die Datei PROG.FOR. Die PIP-Routine wird in der vierten Zeile durch Drücken der RETURN-Taste beendet, und CP/M meldet sich wieder mit dem Bereitschaftszeichen (A>).

Es sollen nun einige Regeln für die Übertragung von Dateien behandelt werden. Das allgemeine PIP-Format hat folgendes Aussehen:

```
d:kopie = d:original
```

Mit „d“ wird das Laufwerkskennzeichen (drive) gesetzt, „kopie“ steht für die neue Bezeichnung der kopierten Dateien und „original“ für die alte Bezeichnung der Originaldatei. Die beiden Laufwerkskennzeichen können auf dasselbe oder auf verschiedene Laufwerke hinweisen. Das Laufwerkskennzeichen der rechten Seite kann weggelassen werden, weil PIP dann annimmt, daß sich die zu kopierende Datei im selektierten Laufwerk befindet. Auf der linken Seite kann das Laufwerkskennzeichen dann weggelassen werden, wenn die Dateibezeichnung angegeben ist. Soll eine Datei mit der gleichen Bezeichnung wie die Originaldatei kopiert werden, ist die Eingabe eines abgekürzten PIP-Befehls möglich:

```
d2: = d1:Original
```

Das Laufwerk d1 muß dabei von d2 verschieden sein. Die PIP-Routine geht in diesem Fall davon aus, daß die Originaldatei mit der gleichen Dateibezeichnung auf die Diskette des mit d2 angegebenen Laufwerks kopiert werden soll.

Durchgeführt wird dieser Befehl jedoch nur bei unterschiedlichen Laufwerkskennzeichen, weil auf einer Diskette nie zwei Dateien mit der gleichen Bezeichnung existieren dürfen.

Mit dem Befehl

```
*B: = A:TEST.INT↵
```

wird die Datei TEST.INT von A auf B kopiert.

Es sollen nun einige Beispiele analysiert werden. Angenommen wird, daß Laufwerk A selektiert ist und sich die Datei DATEI1.DAT auf Laufwerk A und die Datei PROGRAMM.TXT auf Laufwerk B befinden. Sind folgende PIP-Befehle erlaubt?

```
    A>PIP↵
(1) *A: = B:PROGRAMM.TXT↵
(2) *B: = DATEI1.DAT↵
(3) *A:DATEIREV.DAT = A:DATEI1.DAT↵
(4) *A:DATEI1.TXT = DATEI1.DAT↵
(5) *B:DATEI1.DAT = DATEI1.DAT↵
```

Alle oben aufgeführten Befehle sind erlaubt. Der PIP-Befehl (2) entspricht auch

```
B: = A:DATEI1.DAT↵
```

Wenn das Laufwerkskennzeichen weggelassen wird, bezieht sich der PIP-Befehl nur auf Dateien des zugeschalteten (aktuellen) Laufwerks. Die Angabe von A: hätte in Zeile (3) weggelassen werden können.

Die Dateibezeichnung DATEI1.TXT von (4) entspricht nicht der Dateibezeichnung DATEI1.DAT, so daß auch dieser Befehl erlaubt ist. Der Befehl von (5) kann in der abgekürzten Form von (2) geschrieben werden.

Kopieren von Dateigruppen

Dateigruppenbefehle im PIP erlauben das Kopieren von Dateigruppen. Sollen die folgenden Dateien:

```
DATEI1.DAT
BRIEF.TXT
PROGRAMM.INT
```

von B nach A kopiert werden, so ist dies wie folgt erreichbar:

```
A>PIP↵
*A: = B:DATEI1.DAT↵
*A: = B:BRIEF.TXT↵
*A: = B:PROGRAMM.INT↵
*↵
A>
```

In bestimmten Fällen jedoch kann diese Befehlsfolge mit Hilfe der Anwendung von PIP-Gruppensymbolen abgekürzt werden. Für das Kopieren von Dateigruppen werden im PIP die Sondersymbole * und ? bereitgestellt. Das Zeichen ? kann in einer Dateibezeichnung an beliebiger Stelle gesetzt werden, wo es jedes erlaubte Zeichen ersetzt.

So korrespondiert die Bezeichnung

```
DATEI?.DAT
```

z.B. mit den Dateien

DATEI1.DAT
DATEI2.DAT
DATEI3.DAT

jedoch nicht mit der Datei

DATEI44.DAT (ein Zeichen zuviel)

Es sollen folgende Dateien:

DATEI1.DAT
DATEI2.DAT
DATEI3.DAT

von B nach A kopiert werden. Der Befehl lautet dann:

```
A>PIP A: = B:DATEI?.DAT↵
```

Mit diesem einzigen Befehl wird mit Hilfe des Gruppensymbols die Übertragung aller Dateien abgewickelt. Die Directory von Laufwerk B wird dabei so lange durchsucht, bis mit der Gruppenbezeichnung übereinstimmende Dateien gefunden sind. Hierbei muß beachtet werden, daß auch z.B. eine eventuell existierende Datei

DATEIX.DAT

kopiert wird.

Das zweite Gruppensymbol ist das Zeichen *, das noch effektiver eingesetzt werden kann. Es korrespondiert mit allen Zeichen im ganzen Feld unabhängig von der Anzahl der Zeichen. Angenommen B enthielte folgende Dateien:

DATEI1.DAT
DATEI2.DAT
BRIEF.TXT
CBASIC.INT
DATEI1. BAK

Die Gruppenbezeichnung *.DAT bezieht sich dann auf die Dateien

DATEI1.DAT
DATEI2.DAT

und die Gruppenbezeichnung DATEI1.* auf die Dateien

```
DATEI1.DAT
DATEI1.BAK
```

Ebenfalls kann die Gruppenbezeichnung *.* gewählt werden, mit der alle Dateien der Diskette von Laufwerk B angesprochen werden (vgl. nächster Abschnitt).

Sollen alle Dateien vom Dateityp COM von A nach B kopiert werden, so genügt die Eingabe

```
A>PIP B: = *.COM↵
```

Beim Kopieren von Dateigruppen protokolliert PIP jeweils die Datei, die gerade übertragen wird. Beispiel:

```
A>PIP B: = *.BAS↵
COPYING –
PROGRAMM1.BAS
PROGRAMM2.BAS
PROGRAMM3.BAS
A>
```

Neben den Kopierbefehlen zum Kopieren einzelner Dateien und von Dateigruppen können auch Befehle zum Kopieren aller Dateien eingesetzt werden.

Kopieren aller Dateien

Wichtig ist, daß zunächst zwischen dem „Kopieren aller Dateien“ und dem „Kopieren ganzer Disketten“ unterschieden wird. Weil das CP/M-System selbst nicht als Datei abgespeichert ist, muß dieses nämlich über eine spezielle Routine (SYSGEN in Kapitel 2) kopiert werden. Wenn eine Diskette nur Dateien enthält, ist das Kopieren aller Dateien ausreichend. Enthält jedoch die Diskette auch das CP/M-System, werden nur die Dateien, nicht aber das CP/M-System selbst kopiert.

Für das Kopieren aller Dateien kann eine umfassende Dateigruppenbezeichnung verwendet werden. Sollen z.B. alle Dateien vom Laufwerk A auf eine Diskette in Laufwerk B kopiert werden, genügt der Befehl

```
A>PIP B: = A:*.*↵
```

Die allgemeine Form dieses Befehls lautet:

```
PIP d2: = d1:*.*
```

Mit d2 ist wieder das Laufwerk der neuen Diskette und mit d1 das Laufwerk der zu kopierenden Diskette anzugeben. Die Dateigruppenbezeichnung *.* korrespondiert mit allen Dateibezeichnungen (d2 muß dabei von d1 verschieden sein).

Zu beachten ist dabei, daß nur die Dateien des gerade aktiven Benutzerbereichs (wenn nicht mit dem USER-Befehl geändert, ist das der Bereich 0) kopiert werden. Außerdem werden Dateien mit dem System-Attribut ignoriert.

In der praktischen Anwendung wird zum Kopieren einer Diskette der folgende Befehl eingegeben:

```
A>PIP B: = A:*.*[VR]↵
```

Innerhalb der eckigen Klammern hinter der rechten Dateibezeichnung werden bestimmte Befehls-Optionen angegeben. Mit dem Buchstaben V (Verify) wird eine spezielle PIP-Routine selektiert, die die Dateiinhalte von Original und Kopie auf Übereinstimmung überprüft. Mit dieser Routine wird eine hohe Kopiersicherheit erreicht. Die Kopierzeit wird dadurch nur unwesentlich verlängert. Die Option R gibt an, daß Dateien mit dem System-Attribut mit kopiert werden sollen.

Kopieren einer ganzen Diskette

Für viele Computer sind spezielle Dienstprogramme zum schnellen Kopieren ganzer Disketteninhalte verfügbar. In diesem Fall wird vom Original eine identische Kopie erstellt (Spur für Spur), einschließlich eines vorhandenen CP/M-Systems. Beide Disketten müssen physikalisch gleich sein. Daten, die bereits auf der neuen Diskette standen, werden zerstört.

Kopieren mit PIP von Disketten mit nur zwei Laufwerken unter CP/M-80

Angenommen, Sie wollen eine Sicherungskopie erstellen von einer gerade erworbenen Diskette. Wenn Sie mehr als zwei Laufwerke haben, ist das unproblematisch: Sie rufen PIP von Ihrer Systemdiskette in Laufwerk A aus auf und kopieren z.B. von B nach C. Sind jedoch nur zwei Laufwerke vorhanden, wird es schon etwas schwieriger. Folgende Punkte sind dabei zu beachten: Wird die Diskette in dem Ziel-Laufwerk gewechselt, so muß ein Warmstart gegeben werden, da sonst die Diskette einen logischen Schreibschutz bekommt. Ein Warmstart muß in jedem Fall

erfolgen, wenn sich das Aufzeichnungsformat für ein Laufwerk geändert hat (gilt nur für entsprechend ausgerüstete Rechner). Im Augenblick des Warmstarts muß im Laufwerk A eine Systemdiskette (eine Diskette mit dem CP/M auf den Systemspuren) stecken.

Das einfachste Verfahren, um eine Sicherungskopie einer Nicht-Systemdiskette zu machen, besteht darin, diese oder die neue Diskette erst einmal mit einem System zu versehen (z.B. mit SYSGEN). Wird PIP auch auf eine der beiden Disketten kopiert, gibt es keine weiteren Probleme. Ist dies nicht möglich (z.B. aus Platzgründen), so muß PIP von einer dritten Diskette aus gestartet werden:

```
A>PIP↵
*
```

Danach wird diese durch die zu kopierende Diskette ersetzt (das Aufzeichnungsformat muß jedoch übereinstimmen). Mit

```
*B:=A:*.*↵
```

kann jetzt zum Laufwerk B kopiert werden. Vor Abbruch des PIP-Programms muß wieder eine System-Diskette in Laufwerk A eingelegt werden.

```
*↵
A>
```

KOPIEROPERATIONEN ZU GERÄTEN

Einführung

Eine Übertragungsoperation ist zum Beispiel das Ausdrucken einer Datei auf dem Drucker; die Datei wird von der Diskette zum Drucker kopiert. Die PIP-Routine verfügt über allgemein verwendbare Übertragungsmöglichkeiten, die nicht nur die Übertragung von Laufwerk zu Laufwerk, sondern auch zwischen unterschiedlichen Geräteeinheiten ermöglichen.

Die allgemeinen Funktionen sollen im folgenden Abschnitt beschrieben werden. Hierbei wird zunächst die Übertragung zwischen zwei beliebigen Geräteeinheiten des Computers behandelt. Darüber hinaus werden neue erweiterte Funktionen des PIP erläutert, wie z.B. Routinen zur Verkettung von Dateien.

Zunächst soll das Drucken eines Dateiinhaltes besprochen werden.

Ausdrucken einer Datei

Eine Datei kann mit der PIP-Routine oder anderen Programmen ausgedruckt werden. Bei der Anwendung eines Textverarbeitungsprogramms oder eines speziellen Editier- und Druckprogramms sollte dieses auch zum Drucken der generierten und verarbeiteten Dateien verwandt werden. Wenn zum Beispiel Daten über ein Anwendungsprogramm generiert und verwaltet werden, so kann davon ausgegangen werden, daß dieses Programm auch spezielle Routinen zum formatierten Drucken dieser Dateien enthält.

Der wichtigste Vorteil spezieller Druckprogramme ist die Möglichkeit des formatierten Drucks. So können automatische Formatierungen, Tabulatoren, Zeilenvorschübe, Seitennumerierungen und andere Druckaufbereitungen generiert werden.

Für das einfache Ausdrucken von ASCII-Dateien wird im CP/M der TYPE-Befehl zur Verfügung gestellt (vgl. Kapitel 1):

```
A>^P
A>TYPE DATEI.TXT↵
```

Das Steuerzeichen ^P schaltet den Drucker zur Bildschirmausgabe parallel. Mit diesem einfach anzuwendenden Befehl wird Zeile für Zeile auf dem Drucker der gleiche Text wie auf dem Bildschirm ausgedruckt. Dabei wird eine 1:1 Kopie der Datei, so wie diese auf der Diskette gespeichert ist, erstellt.

Der TYPE-Befehl unterstützt dabei nur eine Druckaufbereitungsfunktion, nämlich die Umsetzung der in der Datei vorhandenen Tabulatorsteuerzeichen (CTRL-I). Beim Erkennen eines solchen Steuerzeichens wird der Druckkopf auf die nächste Tabulatorposition weitergeschoben (jeweils acht Spalten).

Der TYPE-Befehl wird hauptsächlich für das schnelle Überprüfen bzw. Erkennen von Dateiinhalten am Bildschirm und für Probeandrucke benutzt. Über den Bildschirm können Texte im allgemeinen mit einer Geschwindigkeit von 9600 Baud im Gegensatz von nur 300 bis 600 Baud am Drucker ausgegeben werden. Aus diesem Grunde ist eine Ausgabe über den Bildschirm in wesentlich kürzerer Zeit als über den Drucker möglich.

Mit dem PIP-Programm können ebenfalls Dateien ausgedruckt werden. Hierbei benutzt PIP die allgemeinen Übertragungsfunktionen.

Übertragung von Dateien

Eine Datei kann zu jeder beliebigen Geräteeinheit gesendet werden. So kann eine Datei z.B. zu einem Diskettenlaufwerk, zum Drucker, zum Bildschirm, zum Lochstreifenstanzer und zum Kassettenrecorder über-

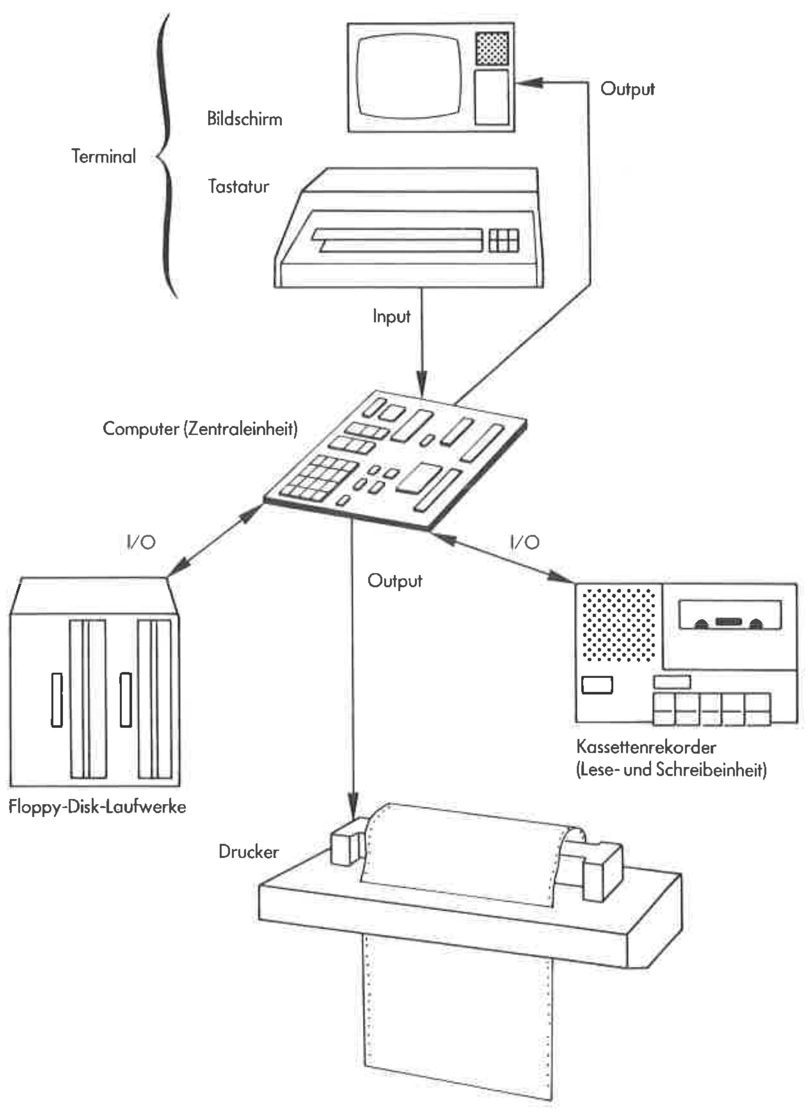


Abb.3.1: Systemkomponenten

tragen werden. Bedingung ist nur, daß keine Eingabeeinheiten angesprochen werden wie z.B. der Kartenleser oder die Tastatur.

Ein Drucker ohne Tastatur kann nur Daten bzw. Dateien empfangen. Ein Drucker mit Tastatur verfügt über die Funktion eines Terminals und ist in der Lage, auch Dateien zu generieren.

Die Abb. 3.1 zeigt, wie ein „Input" (Eingabe) von einer Geräteeinheit empfangen und ein „Output" zu einer anderen Geräteeinheit gesendet werden kann. Alle Informationen und Programme werden dabei über die Recheneinheit übertragen und gesteuert.

Für das Listen einer Datei auf dem Drucker liest der Computer zunächst die Datei von der Diskette (Input) und überträgt sie dann zum Drucker (Output). Die meisten Programme lesen die Dateien blockweise (einen Sektor pro Zeiteinheit). Hierdurch können auch mit kleinen Speicherbereichen große Datenmengen übertragen werden (vgl. Abb. 3.2).

Daten oder Dateien werden auch über die Tastatur eingegeben oder zum Bildschirm gesendet. Genauso können sie zu einem Laufwerk als Laufwerk-zu-Laufwerk-Übertragung transportiert werden. Diese Anforderungen müssen an alle Übertragungsprogramme gestellt werden. In vielen Fällen werden diese Funktionen auch durch spezielle Zusatzprogramme in Anwendungsprogrammen unterstützt. Das PIP-Programm enthält jedoch alle diese allgemeinen Funktionen in seinem Grundbefehlsvorrat. Mit einigen PIP-Befehlen, die mit speziellen Zuweisungsschlüsseln für die anzusprechenden Geräteeinheiten kombiniert werden,

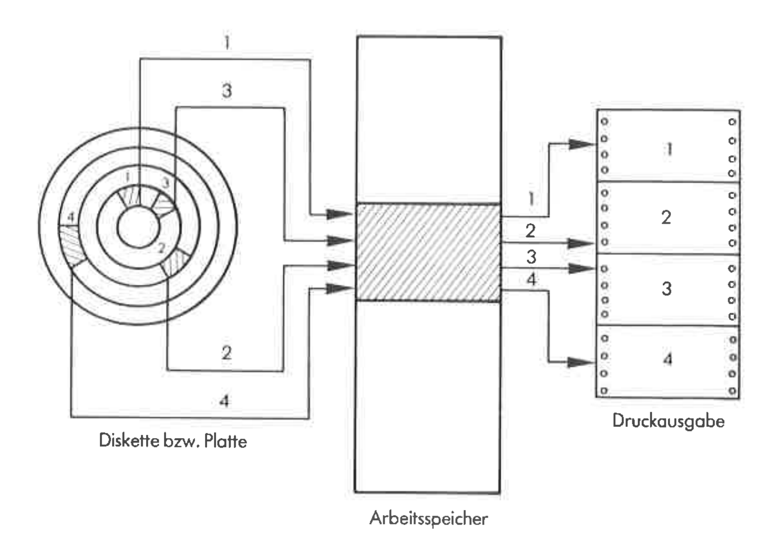


Abb.3.2: Übertragung über den Arbeitsspeicher

können so wirkungsvolle und komplexe Operationen abgewickelt werden.

Für die problemlose Abwicklung von PIP-Routinen müssen die Geräteeinheiten definiert werden. Es sollen deshalb zunächst die PIP-Bedingungen zur Zuweisung einzelner Geräteeinheiten besprochen werden.

Zuweisung der Geräteeinheiten

Im PIP-Befehl werden sowohl logische als auch physikalische Gerätenamen verwendet. Ein physikalischer Gerätename ist der aktuelle Name einer vom System selektierten Geräteeinheit. Ein logischer Gerätename entspricht einem symbolischen Namen für eine Gruppe von Geräteeinheiten, die alle vom System bedient werden können. Zum Beispiel ist LST: der Name der „LIST"-Einheit. Diese Geräteeinheit ist im allgemeinen zwar der Drucker, jedoch kann hier genauso eine andere Geräteeinheit wie z.B. ein Fernschreiber oder ein Modem zugewiesen werden, mit dem Daten über Telefonleitungen geschickt werden können. Der Gerätetyp (z.B. Fernschreiber oder Matrixdrucker) braucht nicht bekannt zu sein, erforderlich ist nur der logische Name.

In PIP-Befehlen sind die folgenden logischen Namen erlaubt:

CON: Für „Console" oder ein Terminal, das Tastatur und Bildschirm besitzt (Eingabe/Ausgabe)
LST: für „List"-Einheiten wie z.B. Drucker (nur Ausgabe)
RDR: Eingabe vom Lochstreifenleser (nur CP/M-80)
PUN: Ausgabe auf Lochstreifenstanzer (nur CP/M-80)

Als Zuordnungsgeräte der logischen Namen RDR: und PUN: werden heute i.a. keine Lochstreifen- oder Lochkarteneinheiten gewählt, weil diese immer weniger genutzt werden. Die RDR:- und PUN:- Zuordnungen werden deshalb vorwiegend für andere Geräteeinheiten zur sequentiellen Verarbeitung (batch) eingesetzt. Beim CP/M-Plus steht dafür die Bezeichnung AUX: (Auxiliary).

Als CON:-Zuordnung wird im allgemeinen ein CRT-Terminal (Cathode Ray Tube Terminal, Bildschirmterminal) mit Tastatur und als LST:-Geräteeinheit ein Drucker verwendet.

Die Übertragungsgeschwindigkeiten zu den verschiedenen Einheiten werden vom CP/M-System verwaltet und vorher bei der Installation (Anpassung des CP/M an die aktuelle Gerätekonfiguration) festgelegt.

Bei der Installation eines CP/M-Systems werden spezielle Routinen für den angeschlossenen Drucker, das Bildschirmterminal und das Disketten- bzw. Plattenlaufwerk mit den entsprechenden Übertragungsgeschwindigkeiten ausgewählt und festgelegt. Werden die Eingabe- oder

Ausgabeeinheiten gewechselt, so muß deshalb auch zwangsläufig die CP/M-Systemdiskette gewechselt werden.

Mit dem STAT-Befehl (vgl. Kapitel 2) können die aktuellen logischen Gerätezuweisungen jederzeit abgerufen werden. Genauso können die Zuweisungen auch mit dem STAT-Befehl wieder geändert werden.

Praktische Übungen

Die logischen Gerätenamen können problemlos auch in PIP-Befehlen verwendet werden:

```
A>PIP↲
*CON: = BEISPIEL.TXT↲
*LST: = B:BEISPIEL.BAK↲
*PROG.BAS = RDR:↲
*PUN: = PROG.BAS↲
*↲
A>
```

Der erste PIP-Befehl sendet eine Kopie der Datei BEISPIEL.TXT (die über das selektierte Laufwerk A erreichbar ist) zur Konsoleinheit (in den meisten Fällen das Bildschirmterminal). In Abbildung 3.3 ist dieser Ablauf schematisch dargestellt.

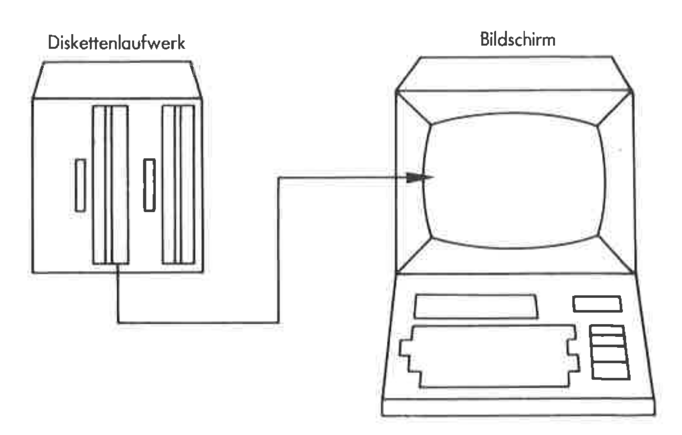


Abb. 3.3: Dateiübertragung zur Konsole: CON: = BEISPIEL.TXT

Der zweite PIP-Befehl überträgt eine Kopie der Datei BEISPIEL.BAK von Laufwerk B zur aktuellen LST:-Einheit (im allgemeinen ein Matrixdrucker oder Teletype – vgl. Abb. 3.4).

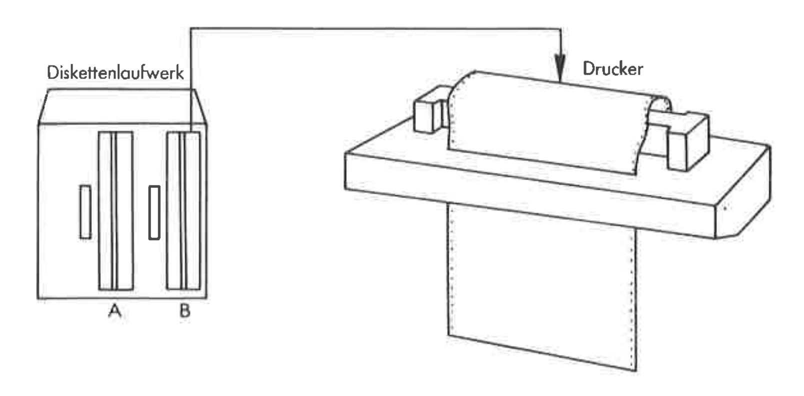


Abb. 3.4: Dateiausdruck: LST: = B:BEISPIEL.BAK

Mit dem dritten PIP-Befehl werden von der Leseeinheit RDR: Daten gelesen und zur Datei PROG.BAS generiert. Diese Leseeinheit ist im allgemeinen ein Programm auf Lochstreifen, Lochkarten oder Kassette, das in das System eingelesen und dann in einer Datei auf der Diskette gespeichert wird (vgl. Abb. 3.5).

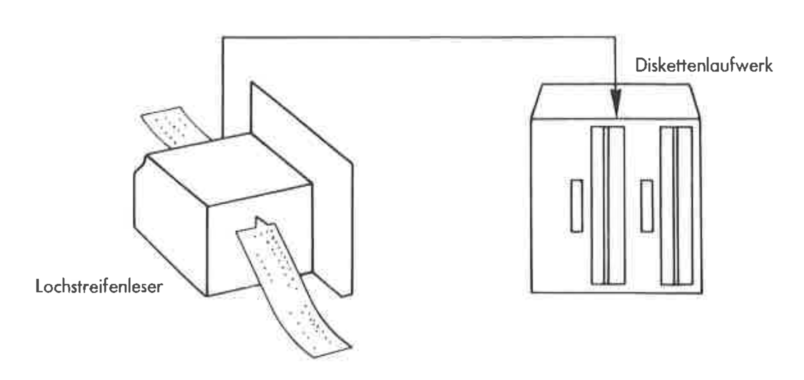


Abb. 3.5: PROG.BAS = RDR:

Die umgekehrte Richtung der Dateiübertragung, nämlich zu einem Lochstreifenstanzer, Lochkartenstanzer oder Kassettenrecorder, wird durch den vierten PIP-Befehl erreicht. Er sendet die Kopie der Datei PROG.BAS zur PUN:-Einheit (vgl. Abb. 3.6).

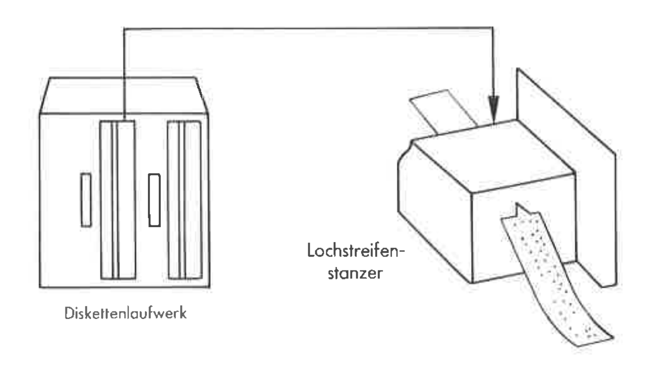


Abb. 3.6: Dateiübertragung zum Lochstreifenstanzer PUN: = PROG.BAS

Physikalische Gerätenamen bei CP/M-80

Die folgenden physikalischen Gerätenamen können in PIP-Befehlen ebenfalls verwendet werden. Sie stellen gültige physikalische Gerätezuordnungen dar:

TTY: für Konsole bzw. Terminal; Leser, Stanzer oder LIST-Einheit (Teletype)
CRT: für Konsole bzw. Terminal; Leser, Stanzer oder LIST-Einheit (Bildschirm)
PTR: für einen Lochstreifen- oder Lochkartenleser
PTP: für einen Lochstreifen- oder Lochkartenstanzer
LPT: für eine LIST-Einheit (Zeilendrucker)
UC1: für eine vom Benutzer definierte Konsole oder Terminal
UR1: für einen vom Benutzer definierten Leser
UR2: für einen zweiten vom Benutzer definierten Leser
UP1: für einen vom Benutzer definierten Stanzer
UP2: für einen zweiten vom Benutzer definierten Stanzer
UL1: für eine vom Benutzer definierte LIST-Einheit

Wichtig: Die Einheit BAT: wurde nicht aufgeführt, da sie nur die Werte für RDR: und LST: zuordnet.

SPEZIELLE KOPIEROPERATIONEN

Einführung

Mit den bisher behandelten Befehlen können alle einfachen Kopieroperationen abgewickelt werden. Im folgenden Abschnitt werden komplexere Übertragungsprozeduren mit Text- und „Hex"-Dateien behandelt. Hierbei werden alle im PIP-Programm enthaltenen Zusatzinformationen ausgenutzt.

Spezielle Gerätenamen

Zur Abwicklung spezieller Übertragungsprozeduren werden im PIP zusätzliche Gerätenamen verwendet. Sie können im PIP-Befehl wie die bisher bekannten Gerätenamen eingesetzt werden:

NUL: sendet 40 Nullen (ASCII-Code) zur Geräteeinheit (im allgemeinen die Stanzeinheit). Im folgenden Beispiel werden nach der Übertragung des Programms PROG.HEX noch 40 Nullen gesendet (d.h. Leerstreifen erzeugt):

```
*PUN: = PROG.HEX,NUL:↵
```

EOF: sendet ein Dateiendezeichen (EOF = ASCII^Z) zur Geräteeinheit. Bei ASCII-Textdateiübertragungen wird dieses Zeichen automatisch gesendet, so daß dieses nur in einigen Sonderfällen benötigt wird. Im folgenden Beispiel:

```
*PUN: = NUL:,X.ASM,EOF:,NUL:↵
```

werden zunächst zur Stanzeinheit 40 Nullen, dann eine Kopie der Datei ASM, ein nachfolgendes EOF-Zeichen (^Z) und weitere 40 Nullen übertragen.

PRN: entspricht der LST:-Einheit mit der Ausnahme, daß die Tabulatorsteuerungen auf alle 8 Spalten festgesetzt, die Zeilen numeriert und die ausgedruckten Seiten über automatische Formularvorschübe (alle 60 Zeilen, wenn nicht als Option anders vereinbart) gesteuert werden:

```
*PRN: = BEISPIEL.TXT↵
```

INP: ist eine spezielle Gerätebezeichnung, die in das PIP-Programm eingefügt werden kann (patched). Die Einfügung muß dann in Assemblersprache im PIP gesetzt werden. PIP empfängt dann das eingegebene Zeichen über den Aufruf einer bestimmten Speicherposition (103H) und legt dieses Zeichen im Speicherbereich ab Position 109H ab (Paritätsbit muß Null sein – Verwendung des Z-Parameters).

OUT: ist genau wie INP eine spezielle Gerätebezeichnung, die ebenso in das PIP noch eingefügt werden kann. PIP ruft die Speicherposition 106H ab und sendet den Inhalt zum Register C.

Wichtig: Die Speicherpositionen 109H bis 1FFH sind im PIP-Bereich nicht benutzt und können deshalb durch Routinen für spezielle Gerätebetreiber ersetzt werden (z.B. mit dem Debugger DDT, der mit dem CP/M- bzw. MP/M-Betriebssystem geliefert wird).

Beispiele für den Einsatz von INP: und OUT:

```
*MODELL.CLK = INP:↵
```

(Eingabe über spezielle Geräteeinheit und Abspeicherung in der Datei MODELL.CLK)

```
*OUT: = MODELL.CLK↵
```

(Übertragung einer Kopie der Datei MODELL.CLK zur speziellen Geräteeinheit)

Übertragung von Textdateien (ASCII) zu Geräteeinheiten

PIP bietet die Möglichkeit der besonderen Behandlung von Textdateien. Das sind Dateien, die Zeichen aus dem genormten ASCII-Zeichensatz (siehe Anhang) beinhalten. Alle üblichen Terminals und Drucker benutzen den ASCII-Zeichensatz zur Kommunikation. Maschinenprogramme, der Code, den der Mikroprozessor versteht, sind natürlich nicht in ASCII codiert. Trotzdem können sie teilweise einen Inhalt haben, der bestimmten ASCII-Zeichen entspricht. Eine Sonderform sind Dateien im HEX-Format, wie sie z.B. der CP/M-Assembler liefert. Hier ist der Maschinencode nach bestimmten Regeln in ASCII-Code übersetzt (hexadezimale Darstellung).

BIT NUMBERS															
b7 →								0	0	0	0	1	1	1	1
	b6 →							0	0	1	1	0	0	1	1
		b5 →						0	1	0	1	0	1	0	1
b7	b6	b5	b4	b3	b2	b1	HEX 1 / HEX 0	0	1	2	3	4	5	6	7
			0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
			0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
			0	0	1	0	2	STX	DC2	"	2	B	R	b	r
			0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
			0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
			0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
			0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
			0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
			1	0	0	0	8	BS	CAN	(	8	H	X	h	x
			1	0	0	1	9	HT	EM	)	9	I	Y	i	y
			1	0	1	0	A	LF	SUB	*	:	J	Z	j	z
			1	0	1	1	B	VT	ESC	+	;	K	[	k	{
			1	1	0	0	C	FF	FS	,	<	L	\	l	|
			1	1	0	1	D	CR	GS	-	=	M	]	m	}
			1	1	1	0	E	SO	RS	.	>	N	^	n	~
			1	1	1	1	F	SI	US	/	?	O	_	o	DEL

NUL Null
SOH Beginn des Kopfes (Start of Heading)
STX Beginn des Textes (Start of Text)
ETX Ende des Textes (End of Text)
EOT Ende der Übertragung (End of Transmission)
ENQ Anfrage (Enquing)
ACK Bestätigung (Acknowledge)
BEL Klingel (Bell)
BS Zurücksetzen (Backspace)
HT Horizontaler Tabulator (Horizontal Tabulation)
LF Zeilenvorschub (Line Feed)
VT Vertikaler Tabulator (Vertical Tabulation)
FF Format-Vorschub (Form Feed)
CR Wagenrücklauf/Zeilenwechsel (Carriage Return)
SO Rückschaltung (Shift Out)
SI Dauerumschaltung (Shift In)

DLE Datenverbindungs-Umschaltung (Data Link Escape)
DC Gerätesteuerung (Device Control)
NAK Negativ-Bestätigung (Negative Acknowledge)
SYN Synchronisations-Leerlauf (Synchronous Idle)
ETB Ende des Übertragungsblocks (End of Transmission Block)
CAN Annullieren (Cancel)
EM Datenträgerende (End of Medium)
SUB Ersetzen (Substitute)
ESC Umschaltung (Escape)
FS Dateitrennzeichen (File Separator)
GS Gruppentrennzeichen (Group Separator)
RS Satztrennzeichen (Record Separator)
US Einheiten-Trennzeichen (Unit Separator)
SP Leerzeichen (Space/Blank)
DEL Löschzeichen (Delete)

Abb. 3.7: ASCII-Konversionstabelle

Die Unterscheidung dieser Dateiformen ist für die Verwendung spezieller Kopierbefehle sehr wichtig. Solche Anwendungen sind z.B. die Ersetzung von Großbuchstaben durch Kleinbuchstaben und umgekehrt, das Löschen einzelner Zeichen während des Kopierens oder das Kopieren einzelner Dateibereiche. Alle diese Funktionen sind nur mit ASCII-Dateien möglich, da das Programm PIP das Zeichen ^Z (CTRL-Z) als Dateiendezeichen interpretiert.

Für das Aneinanderfügen (concatenate) mehrerer Dateien verfügt PIP über einen speziellen Befehl (wird weiter unten näher beschrieben).

Bestimmte Geräte können nur ASCII-Textdateien empfangen oder senden. So kann ein Drucker oder ein Bildschirm nur Textdateien empfangen, andere Geräte wiederum können alle Dateiarten empfangen oder senden.

Die Informationen in einer Datei können auf verschiedene Arten codiert werden. Eine Textdatei ist im allgemeinen im ASCII-Format codiert, bei dem ein 8-bit-Code (ein Byte) für die Repräsentation einzelner Zeichen einschließlich der Steuerzeichen verwendet wird.

PIP überträgt eine Datei bis zur Erkennung eines EOF-Zeichens im ASCII-Code (^Z) oder bis das wirkliche Ende der Datei erreicht ist (ausgenommen hiervon sind beabsichtigte Teilübertragungen von Dateien).

DEZIMAL	BINÄR	HEXADEZIMAL
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Abb. 3.8: Hexadezimaltabelle

Verkettung (Concatenating) von Text-Dateien

Das Zusammenfügen mehrerer Textdateien zu einer einzigen Datei ist eine häufige Anwendung des PIP-Befehls. Hierbei ist darauf zu achten, daß auf der Diskette noch genügend Raum für die Zieldatei ist.

Mit dem folgenden Befehl werden zwei Dateien aneinandergereiht:

```
A>PIP↵
*GESAMT.TXT = TEXT1.TXT,TEXT2.TXT↵
```

Genauso können mehrere Dateien aneinandergereiht werden:

```
A>PIP↵
*GESAMT.ASM = UP1.ASM,UP2.ASM,TEMP.ASM↵
*B:NEU.ZOT = A:ALT.ZAP,B:ALT.ZOT,A:NEU.ZAP↵
*↵
A>
```

In diesem Beispiel werden die Dateien UP1.ASM, UP2.ASM und TEMP.ASM (alle im selektierten Laufwerk A) in der angegebenen Reihenfolge kopiert und in der Datei GESAMT.ASM zusammengefaßt. Der zweite PIP-Befehl fügt die Kopie der Datei ALT.ZAP von Laufwerk A mit der Kopie der Datei ALT.ZOT von Laufwerk B und der Datei NEU.ZAP von Laufwerk A auf Laufwerk B zur Datei NEU.ZOT zusammen.

Die PIP-Routine setzt dabei immer voraus, daß es sich um Textdateien handelt, die alle mit einem EOF-Zeichen abgeschlossen sind. Für das Aneinanderfügen von Dateien, die nicht Textdateien sind, müssen einige Besonderheiten beachtet werden (vgl. Abschnitt „Verketten von Nicht-Textdateien").

Eine andere Möglichkeit, Dateien zu verbinden, ist die Angabe der Datei, an die die andere angehängt werden soll. Mit dem Befehl

```
A>PIP ERSTE.TXT = ERSTE.TXT,ZWEITE.TXT,DRITTE.TXT↵
A>
```

wird der ursprüngliche Dateiinhalt von ERSTE.TXT nicht verändert, sondern um die Dateien ZWEITE.TXT und DRITTE.TXT in dieser Rei-

henfolge vergrößert. Am Beginn der Datei ERSTE.TXT steht nach der Übertragung immer noch der Inhalt der ursprünglichen Datei ERSTE.TXT. Weil die PIP-Routine eine Zwischendatei mit der Bezeichnung ERSTE.$$$ erzeugt, muß auch hier auf genügend Raum auf der Diskette geachtet werden.

Mit den Befehlen zum Zusammenfügen von Dateien können auch mehrere Dateien mit einem Befehl auf der LIST-Einheit ausgedruckt werden. Der Befehl

```
A>PIP LST: = ERSTE.TXT,ZWEITE.TXT↵
```

druckt z.B. die Dateien in der angegebenen Reihenfolge auf dem Drukker aus.

Verketten von Nicht-Textdateien

Zur Markierung des Endes einer Textdatei wird das ASCII-Zeichen EOF (end of file) benutzt; es kann vor dem physikalischen Dateiende liegen. PIP reagiert beim Verketten von Dateien auf dieses Zeichen. Das ist jedoch bei Nicht-ASCII-Daten nicht möglich. Hier muß mit dem O-Parameter die Erkennung des EOF-Zeichens ausgeschaltet werden. Mit

```
A>PIP GESAMT = TEMP1[O],TEMP2[O],TEMP3[O]↵
```

werden die Dateien TEMP1, TEMP2 und TEMP3 kopiert und in der Zieldatei GESAMT zusammengefaßt.

Kopieren von Hexdateien

Dateien im Hexadezimalcode werden im allgemeinen von einem Assembler erzeugt. Ein Assembler übersetzt ein in Assemblersprache geschriebens Quellprogramm in Maschinencode (Folge von Binärzahlen, die bestimmten Instruktionen entsprechen) und speichert diesen als Hexadezimaldatei (Hex-Datei) ab.

Der CP/M-Assembler generiert eine Hex-Datei als Datei mit dem Dateityp „HEX“. Dieser Dateityp hat im PIP eine besondere Bedeutung. Es wird vorausgesetzt, daß diese Datei im „Intel Hex-Format“ vorliegt. Die PIP-Routine überprüft dann automatisch auf vollständiges Format, erlaubte Hexadezimalwerte und Prüfsummen. Aus diesem Grunde sollte der Dateityp „HEX“ vom Benutzer nur in speziellen Fällen vergeben werden.

Soll eine Datei vom HEX-Typ kopiert werden, so können die Parameter H oder I gesetzt werden (für hexadezimale Übertragungen). Bei der Anwendung des Parameters H überprüft die PIP-Routine die Datei nach erlaubtem Intel-Hexadezimalformat. Wird ein Fehler erkannt (z.B. kein Hex-Format), wird am Terminal eine PIP-Meldung ausgegeben. Der Parameter H filtert auch alle nicht notwendigen Zeichen zwischen den HEX-Formaten während des Kopiervorgangs aus.

Der Parameter I setzt automatisch auch den Parameter H (er verfügt über weitergehende Funktionen). Bei der Anwendung des Parameters I werden vom PIP-Programm alle 00-Sätze in der originalen, hexadezimal formatierten Datei (Intel-Hex) ignoriert. Außerdem wird die Datei auf unerlaubte Hexformate untersucht.

Wird von einer Geräteeinheit in eine Datei mit dem Dateityp „.HEX" kopiert, dann überprüft die PIP-Routine die Daten nach erlaubten Hex-Formaten und richtigen Prüfsummen. Wenn also über einen Lochstreifenleser die Datei BEISPIEL.HEX generiert wird, so ist die Angabe des Parameters H überflüssig (für das Überlesen von „00"-Sätzen jedoch wird der Parameter I benötigt).

Falls die PIP-Routine ein ungültiges Format oder einen Prüfsummenfehler entdeckt, wird eine Fehlermeldung zum Terminal gegeben und auf Fehlerkorrekturen gewartet. Wurde z.B. vom Lochstreifenleser eingelesen, kann der Lochstreifen im allgemeinen noch einmal gelesen werden (Lochstreifen noch einmal einlegen). Wenn eine solche Übertragung gestartet werden soll, muß die Return-Taste gedrückt werden, um so der PIP-Routine anzuzeigen, daß die Übernahme erfolgen kann.

Wichtig: Wenn die Geräteeinheit der Leser RDR: ist, kann das Dateiendezeichen (EOF) auch über das Terminal eingegeben werden. Die PIP-Routine liest von der Geräteeinheit, während gleichzeitig die Tastatur nach dem Steuerzeichen für das Beenden der Übertragung abgefragt wird. Im folgenden Beispiel

```
*X.HEX = CON:,Y.HEX[I],PTR:↵

_______________   über das Terminal
_______________   eingegebene Sätze im
_______________   HEX-Format

^Z
*
```

wird eine Datei X.HEX von der „CON:"-Einheit (Terminal) aus bis zur Eingabe von ^Z kopiert. Danach kopiert die PIP-Routine die Datei Y.HEX, wobei vorhandene „00"-Sätze überlesen werden. Zum

Abschluß werden noch Daten vom Lochstreifenleser hinzugefügt, bis ein EOF-Zeichen (^Z) erkannt wird.

Der PIP-Befehl

```
*PROG.X = KLUDGE.HEX[H]↵
*
```

kopiert die Hex-Datei KLUDGE.HEX in die Datei PROG.X und überprüft die Daten während der Übertragung auf ungültige Hexadezimalformate.

OPTIONEN BEI KOPIEROPERATIONEN

Im folgenden Abschnitt sollen einige Verarbeitungsoptionen, die während der Übertragung verfügbar sind, betrachtet werden. Im allgemeinen werden nur einige dieser Optionen genutzt, dennoch ist ein Überblick sehr nützlich. Optionen werden durch Buchstaben in eckigen Klammern hinter der Dateibezeichnung angegeben. (Zwischen dem letzten Zeichen des eigentlichen PIP-Befehls und der eckigen Klammer darf kein Blank stehen!) Sie beeinflussen den Kopiervorgang in bestimmter Weise. Es können dabei auch mehrere Parameter hintereinander gesetzt werden. Einige Parameter benötigen noch zusätzliche Ziffern oder Buchstaben. Sie alle betreffen Kopiervorgänge und können sehr effektiv eingesetzt werden:

A Archivierungs-Option (nicht bei CP/M-80 und CP/M-86). Nur Dateien, die seit dem letzten Kopieren (Archivierung) nicht geändert wurden, werden kopiert. Als Kennzeichnung dient bei den Dateien das Archiv-Attribut.

B Übertragung im „blockmode". Die PIP-Routine schiebt dabei Daten in einen Puffer, bis ein ASCII-„X-off"-Zeichen (^S) erkannt wird. Die Größe des Puffers ist abhängig von der Systemkonfiguration (vgl. spezielles Systemhandbuch). Mit diesem Parameter können fortlaufend Daten von einer sequentiellen Geräteeinheit gelesen werden (z.B. Kassettenrecorder).

```
*SERVE.TXT = RDR:[B]↵
```

C Confirm-Option (nur CP/M-Plus). Bei der Übertragung von Dateigruppen muß jede Kopier-Operation bestätigt werden. Beispiel:

```
*B:=a:*.TXT[C]↵

COPYING –
BRIEF.TXT (Y/N)? Y
LISTE.TXT (Y/N)? N
*
```

Mit Y wird das Kopieren bestätigt. N verhindert das Kopieren.

Dn Delete Option. Die PIP-Routine löscht während der Übertragung die über die n-te Spalte auf dem Bildschirm hinausgehenden Zeichen. Hiermit können lange Zeilen für die Übertragung auf Geräteeinheiten mit kurzen Ausgabezeilen ohne Probleme abgeschnitten werden. Mit dem Befehl

```
*PRN: = LANG.TXT[D40]↵
```

können z.B. Kommentare, die immer an bestimmter Spaltenposition beginnen, ausgeblendet werden.

E Alle Kopiervorgänge werden in der Reihenfolge ihrer Bearbeitung am Terminal gemeldet (echoed). Beispiel:

```
*KOPIE.TXT = QUELL.TXT,S2.TXT,S3.TXT,S4.TXT[E]↵
```

Bei einer großen Folge von Übertragungen können diese leicht verfolgt werden.

F Alle Formularvorschubzeichen (FF = form feed) werden ausgefiltert. Genauso können nachträglich noch Formularvorschubzeichen mit dem Parameter P eingefügt werden. Hiermit ist dann sehr leicht möglich, eine Bereinigung von Dateien vorzunehmen.

Gn Kopiere Dateien aus einem anderen Benutzerbereich (User-Nummer ist n). Diese Option kann (außer bei CP/M-80) auch hinter der Zieldatei angegeben werden und vereinbart den Benutzerbereich, zu dem kopiert werden soll.

H Hexadezimale Datenübertragung. Die PIP-Routine überprüft die Daten auf zulässiges Intel-Hexadezimal-Format.

I Ignorierung von „00"-Sätzen innerhalb der Übertragung von Intel-Hex-Dateien (Parameter H wird automatisch gesetzt). Auch hier werden „HEX"-Dateien erwartet.

L	Umsetzung aller Großbuchstaben (upper case) in Kleinbuchstaben (lower case).
N	Einfügen von Zeilennummern in eine Dateikopie (beginnend bei Zeile 1). Hinter jede Zeilennummer wird ein Semikolon gesetzt. Führende Nullen (z.B. 003) werden weggelassen, es sei denn, es wird der Parameter NZ gesetzt. Bei NZ werden Nullen nicht weggelassen und hinter die Zahlen Tabulatorzeichen gesetzt. Mit dem Parameter T können die Tabulatorstände beliebig verändert und damit sehr gut lesbare Ausdrücke generiert werden.
O	Übertragung von Objektdateien (keine ASCII-Dateien). Die PIP-Routine ignoriert beim Zusammenfassen von Dateien jeweils das physikalische Dateiende (vgl. Zusammenfassen von Dateien).
Pn	generiert nach jeder n-ten Zeile einen Seitenvorschub (intern gesteuerter Seitenvorschub). Wenn n mit 1 oder gar nicht angegeben ist, wird alle 60 Zeichen ein Seitenvorschub erzeugt (vorbestimmt = default). Wenn gleichzeitig auch der Parameter F gesetzt ist, wird von PIP zunächst das Seitenvorschubzeichen in der Datei gelöscht und danach der gewünschte Seitenvorschub erzeugt. Mit dieser Methode kann sehr leicht eine gewünschte Seitenlänge erreicht werden.
Q Zeichenkette ^Z	Die PIP-Routine bricht die Übertragung von einer Geräteeinheit ab, sobald die angegebene Zeichenkette ^Z erkannt wird (eine Zeichenkette oder auch String ist eine Folge einzelner Zeichen; z.B. STRING105%). Der Abschluß der gewählten Zeichenkette wird mit ^Z angegeben. Mit diesem Befehl kann das Kopieren einzelner Dateiabschnitte erreicht werden (vgl. Abschnitt „Kopieren von Dateibereichen“).
R	Dateien mit dem System-Attribut werden mit übertragen.
S Zeichenkette ^Z	Die PIP-Routine beginnt mit der Übertragung von einer Geräteeinheit oder einer Datei, sobald die angegebene Zeichenkette erkannt wird. Die Zeichenkette wird mit ^Z abgeschlossen. Mit diesem Parameter kann sehr leicht ein Dateibereich ab einer bestimmten Startposition übertragen werden.
Tn	Die Tabulatorabstände werden während der Übertragung auf n Spalten gesetzt. Innerhalb einer Textdatei können die Tabulatorsteuerzeichen mit ^I generiert werden. Mit dem Parameter T können so nachträglich andere Tabulierungen erreicht werden.
U	Umsetzung aller Kleinbuchstaben in Großbuchstaben während der Übertragung.
V	Von der PIP-Routine werden alle in die Kopie übertragenen Daten mit der Originaldatei verglichen (verify) und Abweichungen gemeldet.

W Schreibgeschützte Dateien (R/O-Attribut) auf dem Ziel-Laufwerk werden ungefragt überschrieben.

Z Das Paritätsbit eingelesener Daten wird auf Null gesetzt.

Folgende Beispiele zeigen die Anwendung der PIP-Parameter im Zusammenhang:

```
*LST: = BEISPIEL.TXT[NT8P60]↵
```

Dieser Ausdruck sendet die Datei BEISPIEL.TXT zur LIST-Einheit (LST:). Sie wird dabei mit Zeilennummern ausgedruckt (N), Steuerzeichen für den Tabulator werden in Schritten von 8 Spalten interpretiert, und alle 60 Zeilen wird ein Seitenvorschub generiert. Wird anstelle der LST:-Einheit die PRN:-Einheit angesprochen, können die angegebenen Parameter entfallen. Das obige Beispiel verkürzt sich dann auf

```
*PRN: = BEISPIEL.TXT↵
```

Es ist möglich, die vorgegebenen Werte der PRN:-Einheit mit neuen Werten selektiv zu überschreiben:

```
*PRN: = BEISPIEL.TXT[P59]↵
```

Die Datei BEISPIEL.TXT wird mit allen vorgegebenen (default) Parametern (z.B. NT8), ausgenommen dem Seitenvorschub (jetzt 59 Zeilen), zur PRN:-Einheit gesendet.

Mit dem folgenden Ausdruck

```
*LST: = PROG.ASM[NT8U]↵
```

wird das Programm PROG.ASM mit Zeilennummern (N), Tabulatorschritten in jeweils 8 Spalten (T8) und in Großbuchstaben (U) zur List-Einheit übertragen.

Kopieren von Dateibereichen

Bei der Abwicklung allgemeiner Kopieroperationen zur Druckereinheit kann es vorkommen, daß bei langen Ausdrucken diese durch Fehlbedienung (z.B. zufälliges Drücken einer Taste) oder Fehlermeldungen der Druckereinheit (z.B. Papierabriß oder Papierende) ungewollt abgebrochen werden. Das wiederholte Ausdrucken der gesamten Datei kann ohne Probleme, jedoch nur mit hohem Zeitverlust erreicht werden. Für diesen Fall verfügt die PIP-Routine über spezielle Parameter, mit denen ein definiertes Aufsetzen und Absetzen in eine Datei jederzeit möglich ist.

Der PIP-Routine werden hierzu mit Parametern definierte Start- und Stoppzeichenketten mitgeteilt. Der Parameter S gibt die Zeichenkette für den Aufsetzpunkt, bei dem die Übertragung begonnen werden soll, an, und der Parameter Q die Zeichenfolge für den Absetzpunkt, wo die Übertragung beendet werden soll.

Von der PIP-Routine wird dann selbständig der Dateibereich aufgesucht.

Beide Zeichenketten müssen jeweils mit dem Zeichen ^Z(CTRL-Z) abgeschlossen werden. Im folgenden Beispiel wird die Datei bis zu der Stelle kopiert, an der das Wort „Extra" steht.

```
A>PIP↵
*TEIL2 = HAUPT.TXT[QExtra^Z]↵
*↵
A>
```

Sobald die Zeichenfolge „Extra" erkannt wird, bricht die Übertragung ab. Hierbei ist zu berücksichtigen, daß das Wort „Extra" Groß- und Kleinbuchstaben enthält, und daß der PIP-Befehl innerhalb des PIP-Programms und nicht als einzelne Befehlszeile gegeben wurde. Wäre der Befehl als Befehlszeile

```
A>PIP TEIL2.TXT = HAUPT.TXT[QExtra^Z]↵
```

eingegeben worden, wäre das Wort „Extra" mit „EXTRA" erkannt worden, denn wenn die PIP-Routine in einer Befehlszeile gestartet wird, werden alle Zeichen als Großbuchstaben interpretiert! Die Zeichenkette im obigen Beispiel wäre dann nicht gefunden worden.

Das folgende Beispiel zeigt die Kombination des Parameters S und des Parameters Q:

```
A>PIP↵
*EXTRA.TXT = BEISPIEL.TXT[SExtra^ZQAnderes Extra^Z]↵
*↵
A>
```

Nach dem Erkennen der Zeichenkette „Extra“ wird mit der Übertragung der Datei BEISPIEL.TXT begonnen und beim Auffinden der Zeichenkette „Anderes Extra“ abgebrochen. Die Datei EXTRA.TXT enthält einen Teil der Originaldatei zwischen den beiden Zeichenketten „Extra“ und „Anderes Extra“, wobei sich „Extra“ in der neuen Datei befindet, „Anderes Extra“ jedoch nicht mit übertragen wird.

Der PIP-Befehl wurde dabei innerhalb des PIP-Programms eingegeben, um Groß- bzw. Kleinbuchstaben unterscheiden zu können. Sind alle Zeichen in der Datei Großbuchstaben, so können diese mit dem angegebenen Befehl nicht erkannt werden.

Kopieren von Benutzerbereichen

Mit der Angabe des Parameters G können Dateien von einem Benutzerbereich in einen anderen kopiert werden. Dieser Parameter muß nicht angegeben werden, wenn eine Kopie von einer Datei zu einer anderen Diskette unter dem gleichen Benutzerbereich gespeichert werden soll. Wenn z.B. die Datei ORIGINAL im Benutzerbereich 2 von Laufwerk A existiert, so kann die Datei KOPIE ohne Parameter in den Benutzerbereich 2 von Laufwerk B kopiert werden. Soll diese Datei jedoch in einen

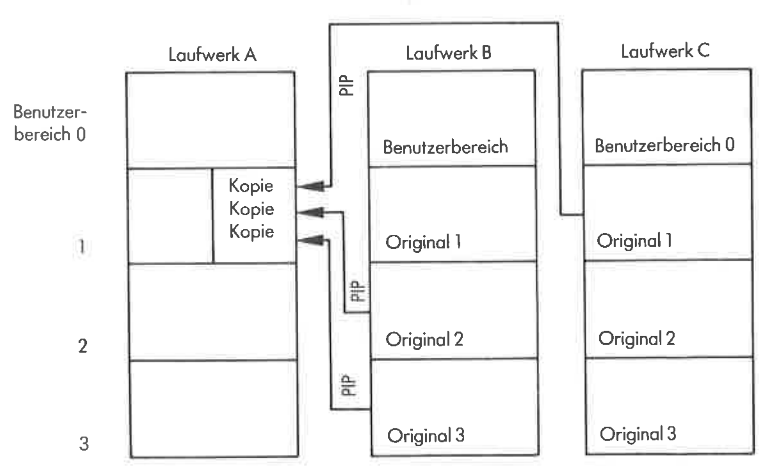


Abb. 3.9: Kopieren mit Umschalten des Benutzerbereichs

anderen Benutzerbereich kopiert werden, so muß der Parameter G angegeben werden.

Ein Benutzerbereich kann mit dem USER-Befehl umgeschaltet werden.

Im folgenden PIP-Beispiel mit der Angabe eines Parameters G

```
*A: = B:JIM.TXT[G3]↲
*↲
2A>
```

wird die Datei JIM.TXT von Laufwerk B und Benutzerbereich 3 nach Laufwerk A in den aktuellen Benutzerbereich kopiert. Der aktuelle Benutzerbereich ist der Bereich 2, wie auch aus der Systemmeldung 2A> nach Beendigung der PIP-Prozedur ersichtlich ist.

Das folgende Beispiel zeigt das Kopieren in einen fremden Benutzerbereich, wie es bei CP/M-Plus, MP/M und CCP/M möglich ist:

```
2A>PIP B:[G3] = A:JIM.TXT↲
```

Bei CP/M-80 ist es jedoch etwas schwieriger, Dateien in einen fremden Benutzerbereich zu bekommen. Es ist hier erforderlich, daß zunächst das PIP-Programm selbst in diesem Bereich steht. Dies kann z.B mit dem SAVE-Befehl dort gespeichert werden. Vorher muß jedoch mit dem DDT-Programm (Dynamic Debugging Tool) das PIP in den Speicher gebracht und sein Umfang bestimmt werden (zu beiden Befehlen vgl. Kap. 2). Im einzelnen ergeben sich folgende Arbeitsschritte:

A>**USER0**↲	Selektieren des Benutzerbereichs 0 (wenn nicht schon selektiert).
A>**DDT PIP.COM**↲	Starten der DDT-Routine mit dem Programm PIP.COM.
DDT VERS.xx.xx	DDT-Meldung.
NEXT PC 1C80 xxxxxx	DDT zeigt die nächste Adresse hinter dem vom Programm PIP.COM belegten Speicherbereich an. Mit dieser Adresse muß die Anzahl der Seiten für den SAVE-Bereich errechnet werden.
–**G0**↲	Rücksprung in das System.

A>**USER3**↵	Umschalten in den Benutzerbereich 3, in den die Kopie von PIP.COM geschrieben werden soll.
A>**SAVE 28 PIP.COM**↵	Die SAVE-Routine generiert eine neue Datei PIP.COM in Benutzerbereich 3 der Diskette von Laufwerk A. Die Datei hat einen Umfang von 28 Speicherseiten und entspricht der Originaldatei PIP.COM. Der Wert 28 errechnet sich aus dem Hexadezimalwert 1C des höherwertigen Byte, wobei C dem Dezimalwert 12 und die Hexziffer 1 dem Dezimalwert 16 entspricht. Die Summe ergibt sich aus 12 plus 16 gleich 28.

Programme, die von Softwarehäusern geliefert werden, befinden sich immer auf dem Benutzerbereich 0 und haben keine Attribute gesetzt.

Nur-Lese-Dateien (read-only)

Wird mit PIP eine Datei kopiert, die auf dem Ziellaufwerk bereits besteht, so wird die alte Datei gelöscht. Ist sie jedoch schreibgeschützt (R/O-Attribut), so antwortet das PIP-Programm mit:

```
A>PIP B:KOPIE = ORIGINAL↵
DESTINATION FILE IS R/O, DELETE (Y/N)?Y
A>
```

In diesem Beispiel wurde versucht, eine Kopie der Datei ORIGINAL mit der Bezeichnung KOPIE zu erstellen. Diese Datei bestand vorher schon mit der gleichen Bezeichnung auf Laufwerk B und hatte ein R/O-Attribut. Dies wurde vom PIP-Programm erkannt und gemeldet als „Zieldatei existiert bereits als R/O-Datei, soll sie gelöscht werden?“ Bei der Eingabe von Y (für YES = Ja) wird die Kopie dann erstellt (Drücken von RETURN ist nicht erforderlich). Wichtig ist, daß die neue Datei KOPIE nicht automatisch auch das R/O-Attribut wieder erhält.

Wird auf die oben erläuterte PIP-Meldung mit N (für NO= Nein) geantwortet, so wird von PIP die Meldung

```
**NOT DELETED**
```

(nicht gelöscht) angegeben.

Soll die Zwischenabfrage im PIP mit der Abfrage, ob die Datei gelöscht werden soll, vermieden werden, so kann der Parameter W verwendet werden. Der Parameter W teilt dem PIP-Programm mit, daß ein mögliches R/O-Attribut ignoriert werden soll. Bei einem Verketten mehrerer Dateien kann der Parameter W am Ende der Befehlszeile angegeben werden:

```
A>PIP GESAMT.TXT = TEIL1.TXT,TEIL2.TXT,TEIL3.TXT[W]↵
A>
```

ZWISCHENDATEIEN BEIM KOPIEREN MIT PIP

PIP ist beim Kopieren von Dateien äußerst vorsichtig. Die Kopie erhält zunächst einmal einen anderen Namen, und zwar wird als Dateityp .$$$ eingesetzt. Erst nach erfolgreicher Übertragung (und Überprüfung, wenn die V-Option gesetzt war) wird eine möglicherweise schon existierende Datei mit dem Zielnamen gelöscht und die .$$$-Datei umbenannt. Diese Arbeitsweise kann u.U. zu Problemen führen, wenn auf dem Ziellaufwerk nur noch wenig freier Speicherplatz ist. In diesem Fall sollte die bereits auf dem Ziellaufwerk bestehende Datei vor der PIP-Operation mit ERA gelöscht werden.

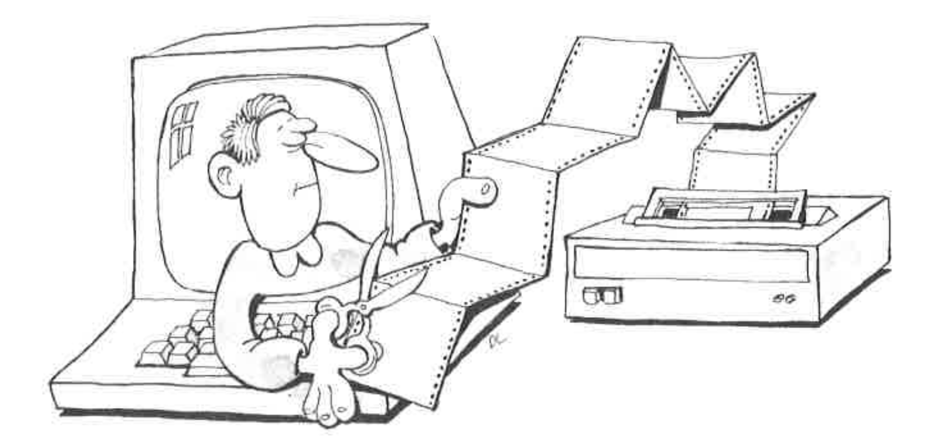
ZUSAMMENFASSUNG

PIP ist ein leistungsfähiges Programm zur Abwicklung allgemeiner Dateiübertragungen. Obwohl die meisten CP/M-Anwender die PIP-Routine nur für die Dateiübertragung zwischen einzelnen Diskettenlaufwerken benutzen, deckt PIP ein breites Spektrum von Funktionen ab:

- Übertragung von Dateien zwischen Disketten-, Plattenlaufwerken und anderen externen Geräten
- Dateigruppenübertragungen und Dateikonzentrationen
- Dateiverarbeitung, -überprüfung und -formatierung

Die PIP-Routine kann sowohl auf Teile einer Datei als auch auf mehrere Dateien angewendet werden. Sie kann zur Bereinigung von Disketten und zur Seiten- und Zeilenformatierung von Dateiausdrucken verwendet werden.

Kenntnisse über die verfügbaren PIP-Parameter bieten darüber hinaus viele weitere Einflußmöglichkeiten und große Vorteile. Jeder CP/M-Anwender sollte daher dieses Kapitel sorgfältig lesen und die beschriebenen Möglichkeiten im Detail anwenden.



Kapitel 4

Anwendung des Editors

EINLEITUNG

Kapitel 1 und 2 sind so aufgebaut, daß alle Anfangsaufgaben bei der Nutzung eines CP/M-Systems abgedeckt werden. Im Kapitel 3 wurde das wichtigste Dienstprogramm, das PIP-Programm, beschrieben. In diesem Kapitel soll ein weiteres bedeutendes Anwendungsprogramm, das mit dem CP/M-Betriebssystem geliefert wird, besprochen werden: der Editor. Es werden die wichtigsten Funktionen des Editors und die Datenübertragungen zwischen Diskette, Arbeitsspeicher und Bildschirmterminal besprochen.

Zunächst ist die Kenntnis der im Detail besprochenen ED-Befehle nicht wichtig, da im Anhang alle Befehle übersichtlich geordnet und nachschlagbar sind. Wichtig ist jedoch, wie die einzelnen Editor-Operationen ablaufen und welche Möglichkeiten der Editor bietet. Wenn dieses Ziel beachtet wird, dann ist ein Einarbeiten in ähnliche Texterstellungsprogramme jederzeit leicht möglich. So macht danach die Anwendung eines Textverarbeitungsprogramms keine besondere Mühe mehr.

Dieses Kapitel ist zwar sehr nützlich, jedoch nicht unbedingt für das Arbeiten mit Mikrocomputersystemen notwendig, weil für die Textbearbeitung auch andere Programme bereitgestellt werden (z.B. WordStar von MicroPro).

WAS IST EIN EDITOR-PROGRAMM?

Mit einem Editor können Textdateien wie Programme Briefe, Abhandlungen, Angebote, Geschäftsformulare oder andere Folgen beliebiger Zeichen erstellt und verändert werden. Das Programm sollte mit überschaubaren Befehlen das Durchsuchen und die Anzeige ganzer Texte, die Veränderung einzelner Zeichen durch Löschen und Überschreiben in einem Arbeitsgang ermöglichen. Genauso sollte es möglich sein, ganze Zeichengruppen herausfinden und ersetzen zu können. Darüber hinaus eignet sich ein guter Editor auch für das Mischen einzelner Dateien.

Wenn mit einem Editor Text eingegeben wird, dann geschieht dies in ähn-

licher Weise wie mit einer Schreibmaschine über die Eingabe einer Textzeile und das Abschließen dieser Textzeile durch Drücken der RETURN-Taste (Wagenrücklauftaste).

In Zukunft wird auch die Texterstelluing mit Mikrocomputern weiter verbessert werden; dennoch soll auf den CP/M-Editor näher eingegangen werden.

Der CP/M-Editor ist ein zeilenorientierter Editor und nicht so leicht anzuwenden wie die heute üblichen Editoren, die Bildschirm-orientiert sind und über Cursor-Funktionen verfügen. Soll mit einem Mikrocomputersystem vorwiegend Text generiert und verändert werden, so ist der Einsatz eines leistungsfähigeren Editors empfehlenswert. Für die Betriebssysteme CP/M und MP/M sind am Markt einige sehr leistungsfähige Textbearbeitungsprogramme erhältlich.

Ein Textbearbeitungsprogramm enthält im Gegensatz zu einem Texterstellungs- und Textveränderungsprogramm noch Zusatzfunktionen für die Druckausgabe (Unterstreichen, Randausgleich, Tabulatorfunktionen und verstärkter Druck). Beim Kauf solcher Textbearbeitungsprogramme sollten jedoch die einzelnen Funktionen sehr genau analysiert werden. So werden Programme als Textbearbeitungsprogramme angeboten, die sich bei der Nutzung allenfalls nur für die Textausgabe eignen (Textformatierer).

DER EDITOR ED

Das Editorprogramm ED.COM (bzw. ED.CMD bei der 86er-Familie) ist im allgemeinen auf der Systemdiskette untergebracht und kann durch die Eingabe von ED und einer nachfolgenden Dateibezeichnung gestartet werden. Soll z.B. die Datei BEISPIEL.TXT generiert werden, so muß

```
A>ED BEISPIEL.TXT↵
```

eingegeben werden. Wird nur ED eingegeben, so erfolgt eine Fehlermeldung, und der Editor kehrt nach CP/M zurück. Die Angabe einer Dateibezeichnung ist also zwingend.

Wird die hinter ED angegebene Datei nicht gefunden, nimmt das Editorprogramm an, daß diese Datei neu generiert werden soll. Diese Datei wird dann zur „Quelldatei“. Spätere ED-Befehle legen den „Quelltext“ in den Textpuffer ab (vgl. Abb. 4.1).

Der Puffer ist ein Speicherbereich innerhalb des Computersystems, der für die Textverarbeitung mit dem Editor reserviert ist. Wenn der Text

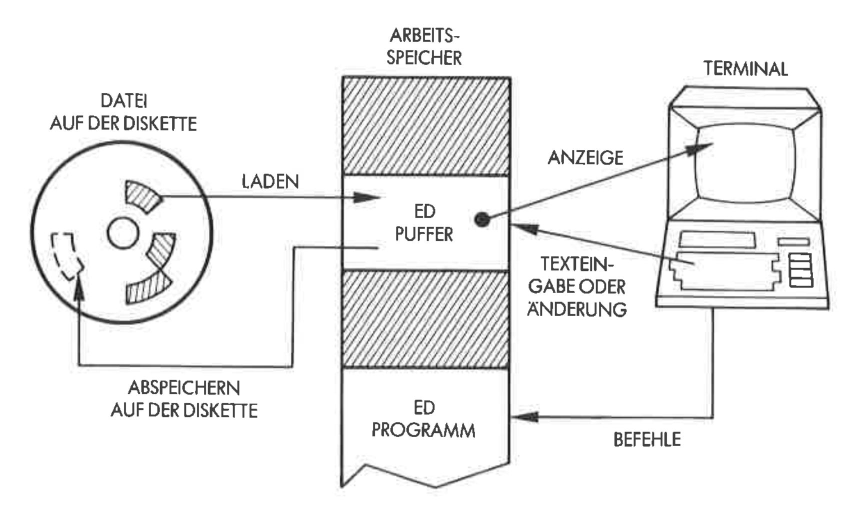


Abb. 4.1: Textpuffer des ED-Editors

sehr lang ist, kann immer nur ein Teil auf einmal geladen werden (diese Beschränkung existiert z.Z. nur im CP/M-Editor, aber nicht bei den meisten anderen Editoren). Das erforderliche Füllen und Leeren des Puffers im Arbeitsspeicher liegt in der Verantwortung des Anwenders!

Neuer Text kann nur innerhalb des Textpuffers generiert oder verändert werden. Der Textpuffer wird jedoch nicht automatisch zur Diskette zurückgespeichert. Wenn der Editiervorgang im Editor abgebrochen wird (oder das System abgeschaltet wird), ohne den Text im Textpuffer zu sichern, geht der Text unwiederbringlich verloren. Weil der ED-Editor Text nur in den Editierpuffer kopiert, ist der Originaltext zwar noch vorhanden, der neu erstellte Text jedoch verloren. Deshalb sollte bei (zeitlich) langen Editierarbeiten häufiger der Text gesichert werden. Abgespeichert und damit gesichert wird der Inhalt des Textpuffers mit dem ED-Befehl E (vor Beendigung des Programms) oder H (ohne Beendigung des Programms). Diese Befehle kopieren zusätzlich noch den Rest der alten Quelldatei in die neue Quelldatei (wird noch näher erläutert).

Die meisten ED-Befehle bestehen aus speziellen Buchstaben mit einer Zahl oder einem Symbol, das die Menge angibt. Diese Befehle werden genau wie alle anderen CP/M-Befehle mit RETURN abgeschlossen.

Andere Befehle sind spezielle Steuerbefehle (wie ^Z und ^C), die bei der Eingabe sofort ausgeführt werden und kein RETURN benötigen.

Das ED-Programm meldet sich mit dem Bereitschaftszeichen *:

```
*
```

Der E-Befehl wird wie folgt eingegeben:

```
*E↵
```

Einige Befehle wirken auf den Text im Editierpuffer, während andere die Übertragung von und zum Editierpuffer veranlassen. Der Editierpuffer kann mit folgender Darstellung verdeutlicht werden:

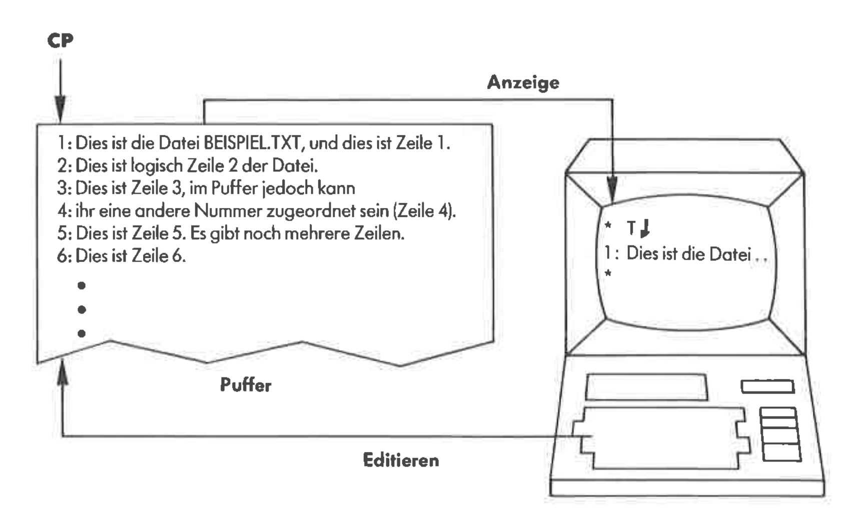


Abb. 4.2: Textbearbeitung

DER ZEICHENZEIGER UND DIE ZEILENNUMMERN

CP in der obigen Darstellung bildet den „Zeichenzeiger" (CP = character pointer). Er wird zwar nicht direkt am Bildschirm angezeigt, jedoch zur Steuerung des ED-Programms benötigt und durch ED-Befehle weiterbewegt.

Der Zeiger zeigt immer auf ein einzelnes Zeichen, und die ED-Befehle wirken dann im allgemeinen auf die dem Zeiger nachfolgenden Zeichen (einschließlich des Zeichens, auf das der Zeiger deutet). Das heißt, daß beim Vorwärtsblättern im Text der Zeiger nach rechts und beim Rückwärtsblättern nach links verschoben wird. Einige Beispiele in diesem Kapitel machen dies noch deutlicher.

Zusätzlich zum „imaginären" Zeichenzeiger (CP) wird noch auf eine Zeilennummer, die nicht Teil des Textes ist, zurückgegriffen. Die Zeilennummern sind genauso wie der Zeichenzeiger nur für den Editierpuffer von Bedeutung. Sie erscheinen nicht beim Ausdruck einer Datei.

VERÄNDERUNGEN EINER TEXTDATEI MIT DEM ED-EDITOR

Für das folgende Beispiel wird davon ausgegangen, daß die Datei BEISPIEL.TXT bereits existiert und den in Abbildung 4.3 angedeuteten Inhalt aufweist.

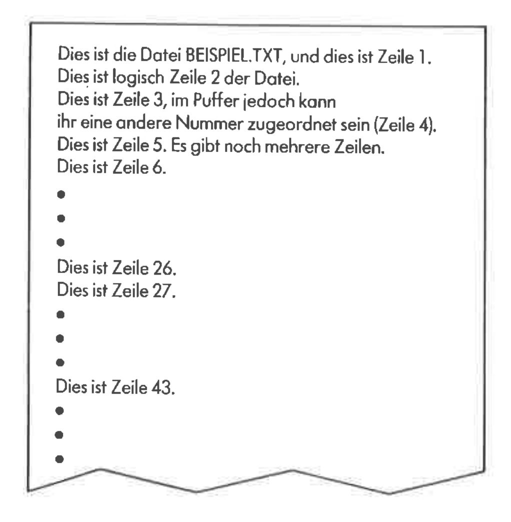


Abb. 4.3: Beispiel einer Datei im Puffer

Bei der Ausführung des CP/M-Befehls

```
A>ED BEISPIEL.TXT↵
```

setzt das ED-Programm einen Bereich im Arbeitsspeicher (transient program area) als Textpuffer fest, generiert eine als Output verwendete Übergangsdatei mit der Bezeichnung BEISPIEL.$$$ (damit die editierte Datei ohne Zerstörung der alten Datei generiert werden kann) und bereitet die Datei BEISPIEL.TXT für die Übernahme in den Textpuffer vor. Die Übernahme (Kopie in Textpuffer) wird dann bei Eingabe des Befehls

A (append) gestartet, worauf die angegebene Anzahl Zeilen im Editierpuffer abgelegt wird (vgl. Abb. 4.4).

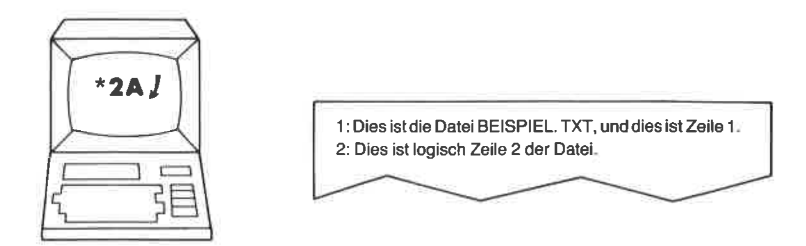


Abb. 4.4: Der Append-Befehl A

Mit dem ED-Befehl 2A werden die ersten beiden Zeilen der Textdatei in den Textpuffer geholt. Sollen mehr Zeilen übernommen werden, muß ein anderer A-Befehl eingegeben werden (vgl. Abb. 4.5).

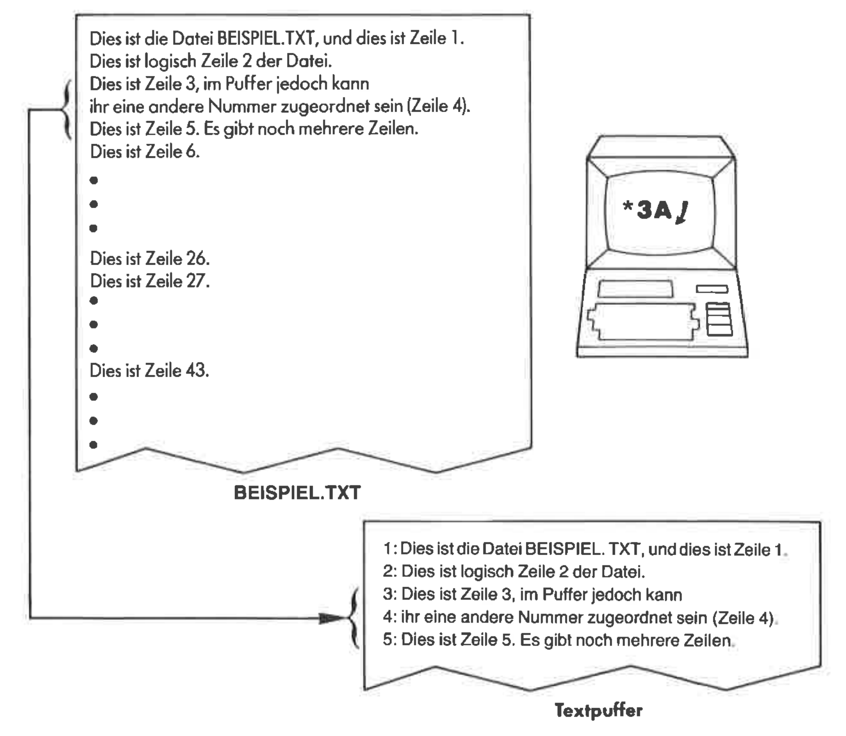


Abb. 4.5: Einfügen von drei weiteren Textzeilen

Danach können diese Textzeilen im Textpuffer verändert und neue Zeilen über die Tastatur hinzugefügt werden (vgl. Abb. 4.6). Beide Vorgänge, das Einladen von Textzeilen aus der Datei und das Einfügen von Textzeilen über die Tastatur, zeigt Abb. 4.6. Genauso ist das Laden einer gesamten Textdatei in den Editierpuffer möglich. Hierbei muß jedoch beachtet werden, daß nur ein begrenzter Speicherraum im Rechner als Textpuffer zur Verfügung steht. Deshalb empfiehlt sich das Ablegen editierter Zeichen in einer Übergangsdatei (temporary output file). Diese Übergangsdatei ist im Editor ED intern mit BEISPIEL.$$$ bezeichnet. Neu editierter Text wird in diese Übergangsdatei entweder mit dem W-Befehl (write), mit dem E-Befehl (exit) oder mit dem H-Befehl (end and reopen) geschrieben. Abb. 4.8 zeigt die Anwendung des W-Befehls.

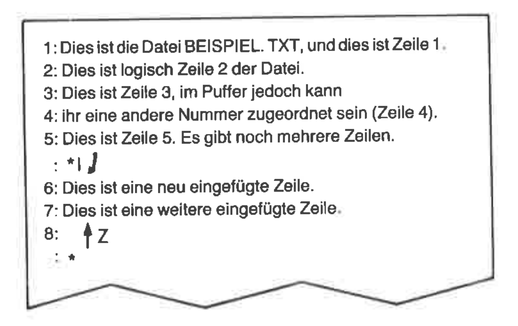


Abb. 4.6: Hinzufügen von zwei neuen Zeilen über die Tastatur

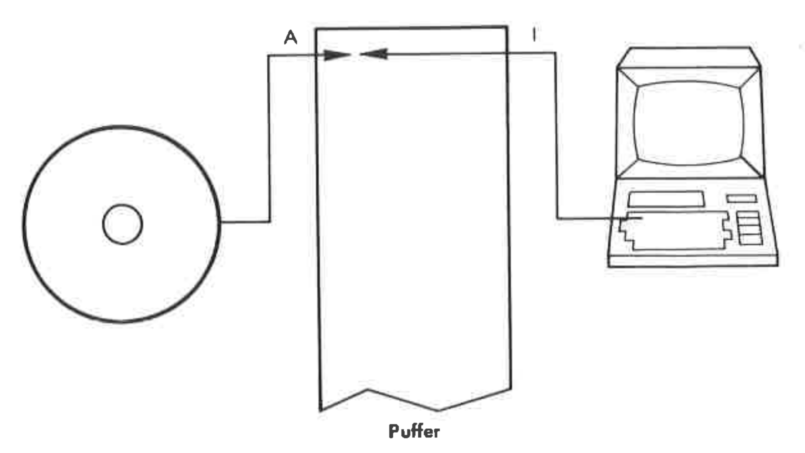


Abb. 4.7: Eingabe von Textzeilen in den Puffer

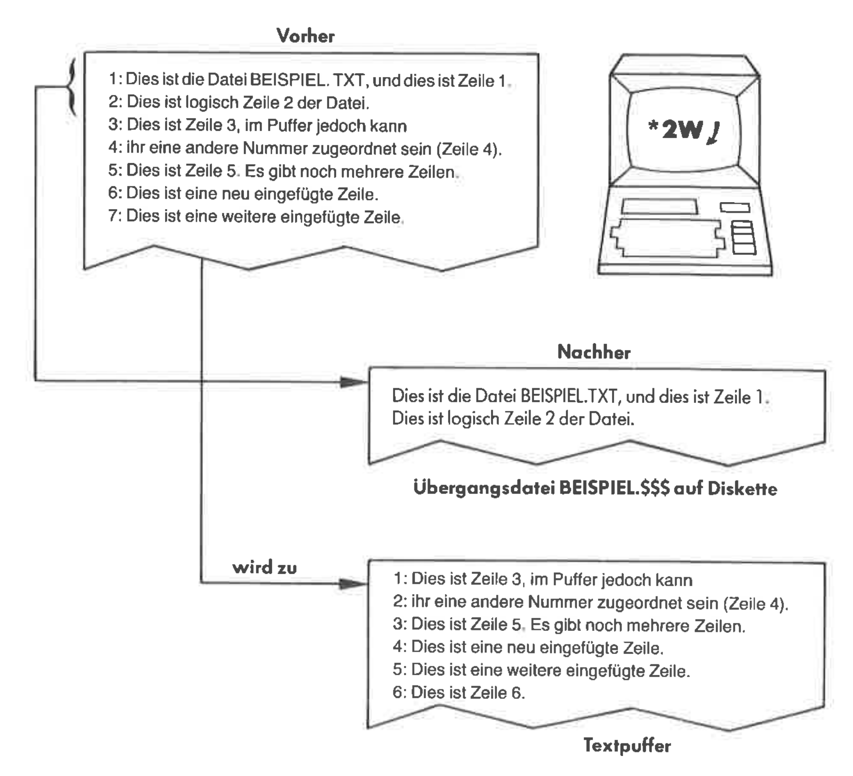


Abb. 4.8: Sichern des Pufferinhalts auf Diskette

Das Beispiel von Abb. 4.8 zeigt den Schreibvorgang für zwei Zeilen. Die Zeilen im Textpuffer werden nach dem Schreibvorgang an den Anfang geschoben, damit mehr Platz für weitere Eingaben geschaffen wird. Im nächsten Beispiel werden die nächsten 20 Zeilen geladen.

Nach der Veränderung des Textes im Textpuffer kann der Editiervorgang abgebrochen werden, indem der Rest des Textpuffers zur Ausgabedatei mit dem E-Befehl geschrieben wird (Abb. 4.9). Mit dem E-Befehl wird auch der Rest der Quelldatei (die Zeichen, die noch nicht in den Textpuffer geladen wurden) zur Ausgabedatei kopiert.

Wichtig ist, daß die Übergangsdatei alle Textzeilen in ihrer vorher definierten Reihenfolge enthält.

Sobald der Inhalt des Editierpuffers ordnungsgemäß in die Übergangsdatei BEISPIEL.$$$ kopiert ist, wird vom ED-Programm zunächst die Datei BEISPIEL.TXT in BEISPIEL.BAK und danach die Datei BEISPIEL.$$$ in BEISPIEL.TXT umbenannt. Hierbei generiert das ED-

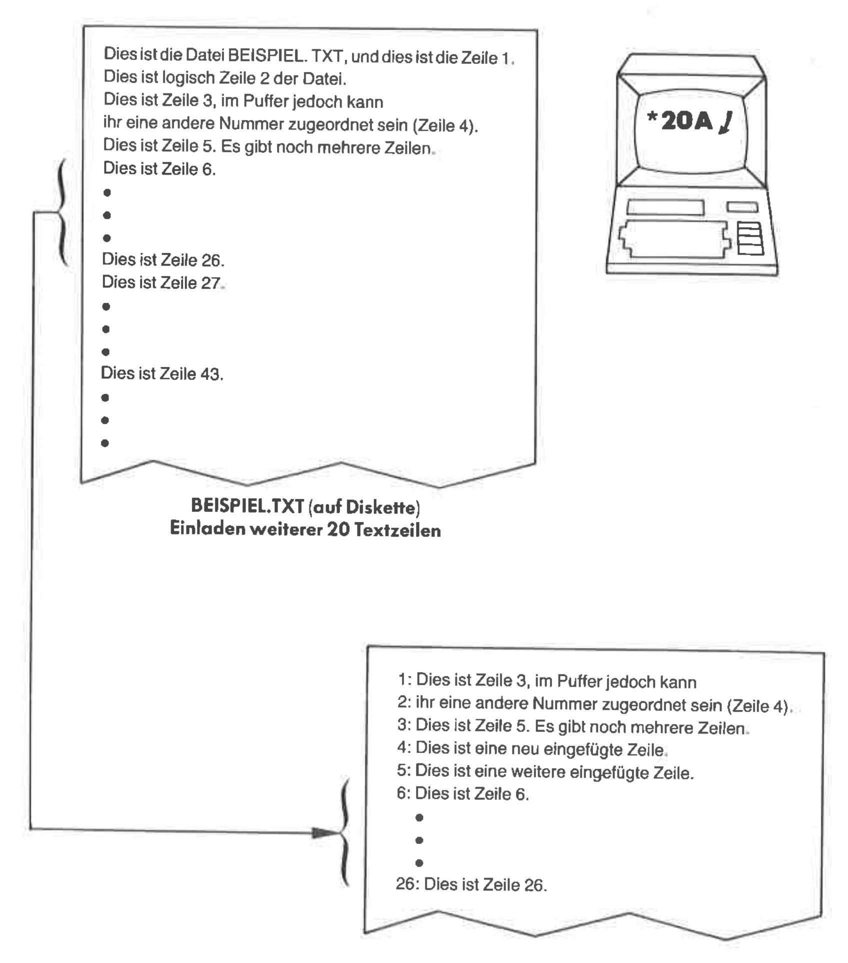


Abb. 4.9: Laden von 20 Textzeilen in den Puffer

Programm eine „Back-up"-Kopie des Originals BEISPIEL.TXT (mit BEISPIEL.BAK bezeichnet) und benennt die modifizierte Datei BEISPIEL.$$$ in BEISPIEL.TXT um (vgl. Abb. 4.10).

Aus diesem Grunde wird auch nicht die Originaldatei auf der Diskette, sondern nur deren Kopie im Textpuffer verändert.

Wichtig: Wenn das ED-Programm auf eine Datei angewendet wird, wird automatisch die entsprechende „BAK"-Datei gelöscht. Verhindern kann man dies nur mit der Unterbrechung des Editiervorgangs mit dem O- oder dem Q-Befehl.

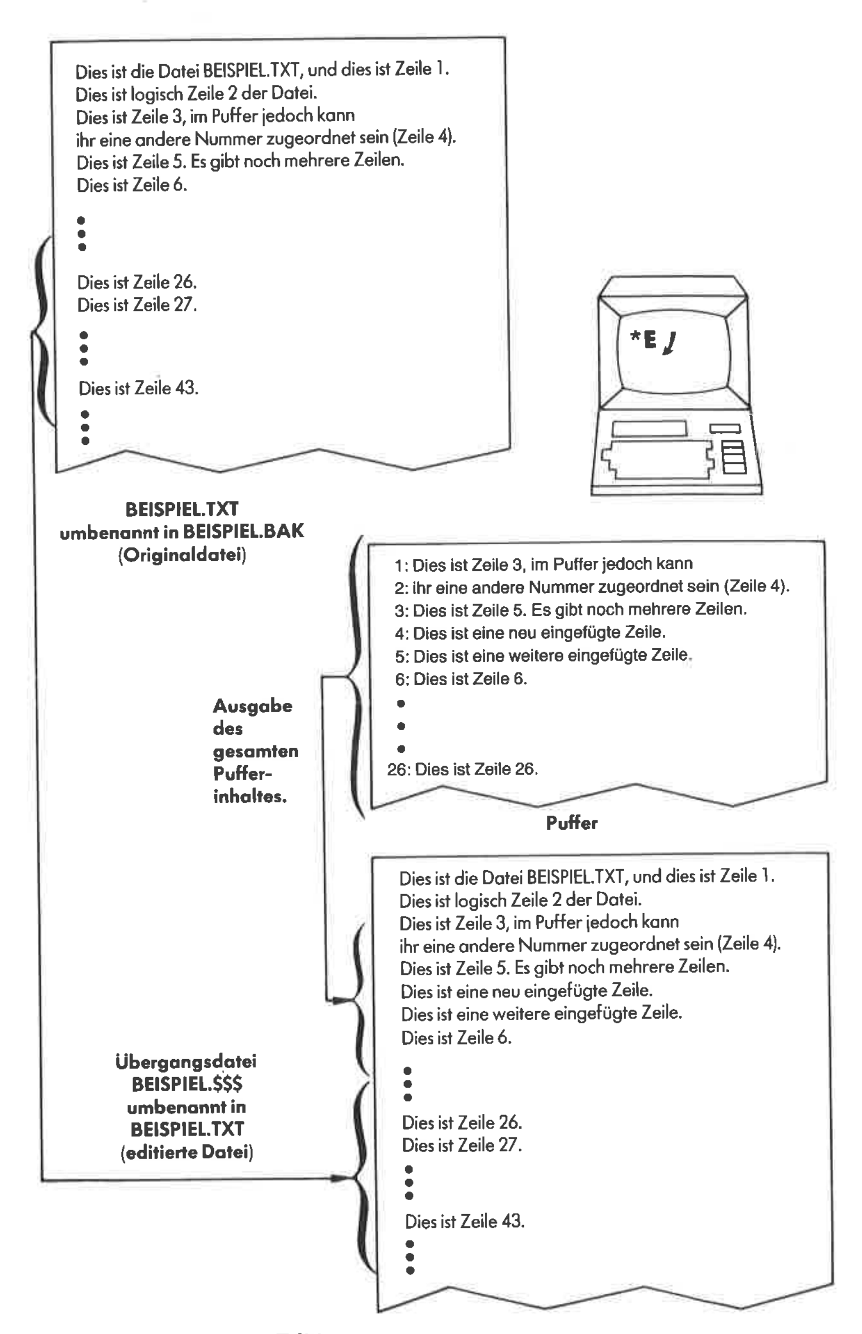


Abb. 4.10: Beenden eines Editiervorgangs

DATEI-MANAGEMENT

Wenn im ED-Befehl Dateibezeichnungen vergeben werden, deren Dateien noch nicht vorhanden sind, so werden diese als neue Quelldateien generiert. Werden Textzeilen in den Textpuffer geladen (mit dem A-Befehl), so geschieht dies vom Beginn der Datei aus. Die Anzahl der Zeilen entsprechen der mit dem A-Befehl angegebenen Zahl. Beim Schreiben von Textzeilen aus dem Puffer in die Übergangsdatei (Dateiname.$$$) wird vom ED-Programm wieder freier Raum für weitere Textzeilen geschaffen. In die Übergangsdatei werden die Textzeilen dabei in der Reihenfolge der Schreibvorgänge geladen, so daß sie dort in gewünschter Anordndung angefordert werden können. Soll der Editiervorgang abgeschlossen werden, wird hierzu der E-Befehl (oder H-Befehl) eingegeben. Vom ED-Programm wird dann, wie oben beschrieben, die Originaldatei (die Quelldatei) in eine „BAK“-Datei umbenannt und die Übergangsdatei (z.B. Beispiel.$$$) mit dem Namen der Originaldatei belegt (z.B. BEISPIEL.TXT). Die angesprochene Datei enthält nun den modifizierten Text.

UNBEABSICHTIGTER ABBRUCH DES EDITIERPROGRAMMS

Wenn das ED-Programm ohne Eingabe des E- oder H-Befehls ungewollt abgebrochen wird (z.B. durch Systemfehler, durch Drücken von ^C oder Netzausfall), so verbleibt die Übergangsdatei mit der Bezeichnung BEISPIEL.$$$ auf dem Datenträger und die ursprüngliche Quelldatei (z.B. BEISPIEL.TXT) wird nicht verändert.

Die „$$$“-Datei enthält nur den Text, der bis dahin weggeschrieben wurde. Der Inhalt des Textpuffers ist verloren, und die Originaldatei besteht in unveränderter Form (bevor der ED-Befehl eingegeben wurde).

Wichtig: Die Eingabe des Befehls Q (Quit) hat die gleiche Wirkung.

Im ED-Programm können Textzeilen einer anderen Textdatei mit bereits im Textpuffer erstellten Textzeilen gemischt werden. Hierzu wird der R-Befehl benötigt, der noch erläutert wird. (Die aufgerufene Datei wird dabei nicht verändert.)

EIN TEXTBEISPIEL MIT DEM EDITOR

Vor dem Generieren einer neuen Textdatei muß zunächst eine neue Dateibezeichnung festgelegt werden (eine Dateibezeichnung, die noch nicht existiert). Im folgenden Beispiel wird die Bezeichnung QUOTE.TXT gewählt. Die Eingabe des folgenden ED-Befehls generiert die Datei QUOTE.TXT:

```
A>ED QUOTE.TXT↵
```

In diesem Beispiel wurde der ED-Befehl von Laufwerk A aus eingegeben und die Datei somit auch auf Laufwerk A generiert. Genauso kann der ED-Befehl auch von anderen Laufwerken durch Angabe des entsprechenden Laufwerkskennzeichens gestartet werden (vgl. ED-Beispiel von Kapitel 1). Soll die Datei auf Laufwerk B generiert werden, so muß vor die Dateibezeichnung das Laufwerkskennzeichen „B:" gesetzt werden.

Das ED-Programm meldet sich mit „NEW FILE" (Neue Datei), wenn eine neue Datei generiert wird. Ist die ED-Routine für einen ED-Befehl bereit, so meldet sie sich mit dem Bereitschaftszeichen *. Danach kann jeder ED-Befehl eingegeben werden. Für das Einfügen eines neuen Textes von der Tastatur aus ist dies z.B. der I-Befehl. Der I-Befehl schaltet auf Einfügen von Text direkt hinter der CP-Position um. Wurde der CP nicht mit einem der verfügbaren ED-Befehle im Text weiterbewegt, so steht dieser am Beginn des Textpuffers. Der Eingabebetrieb beginnt nach der Eingabe von

```
*I↵
```

Der gewünschte Text wird danach genauso wie bei einer Schreibmaschine fortlaufend eingegeben. Abgebrochen wird der Einfügebetrieb durch gleichzeitiges Drücken der CTRL- und der Z-Taste (^Z).

Nach der Eingabe von „I↵" springt der Cursor (blinkendes Einzelfeld oder sonstiges festgelegtes Symbol) zur nächsten Zeile. Von dort aus kann Text eingegeben werden, der automatisch in den Textpuffer Zeile eingefügt wird. Jede Zeile muß mit einem RETURN (CR, carriage return) abgeschlossen werden (entspricht dem Wagenrücklauf einer Schreibmaschine). Hierdurch wird der Zeile eine Zeilennummer im Textpuffer zugeordnet. Wenn eine Zeile im Puffer eingegeben werden soll, die länger als eine Zeile des Bildschirmterminals ist, so muß an der gewünschten Stelle ein ^E eingegeben werden. Der Cursor springt zur nächsten Zeile, ohne daß die Zeile sofort vom Textpuffer übernommen wird. Erst durch Drücken der Return-Taste wird die Zeile im Textpuffer abgelegt. Solche langen Zeilen dürfen aber nicht mehr als 128 Zeichen aufweisen.

Als Löschbefehle können von der Tastatur aus die Tasten RUBOUT (DELETE) und ^U (Löschen einer ganzen Zeile) und als Hilfsbefehle die Steuerbefehle ^E (Wagenrücklauf ohne Übertragung der Textzeile) und ^R (nochmaliges Anzeigen der zuletzt eingegebenen Zeile) verwendet

werden. Darüber hinaus kann auf Bildschirm-Terminals auch die Taste BACKSPACE bzw. ^H (Rücksprung und Löschen eines einzelnen Zeichens) und ^X (Rücksprung zum Anfang der Zeile) als Hilfsbefehle benutzt werden.

Im folgenden Beispiel

```
A>ED QUOTE.TEXT↵
NEW FILE
*I↵
"Wir werden nicht nachlassen in unserem Forschen."
T.S. ELIOT
^Z
*
```

wird mit der Kombination CTRL und Z der Einfügebetrieb abgebrochen. Es wurden zwei Textzeilen erstellt: die erste als Aussage und die zweite als Unterschrift.

Wenn vor der Eingabe des I-Befehls der U-Befehl eingegeben wird, werden alle folgenden eingegebenen Zeichen in Großbuchstaben (upper case) umgewandelt (unabhängig davon, ob der eingegebene Text aus Klein- oder Großbuchstaben besteht). Abgeschaltet wird die automatische Konversion von Klein- in Großbuchstaben durch die Eingabe von –U.

Jetzt kann der eingegebene Text angezeigt werden. Zuerst muß hierzu der CP (Zeichenzeiger) auf den Anfang des Textpuffers gesetzt werden, da dieser während des Einfügebetriebs immer weiter nach rechts geschoben wurde. Abb. 4.11 zeigt die Zeigerposition nach der Eingabe des Textes:

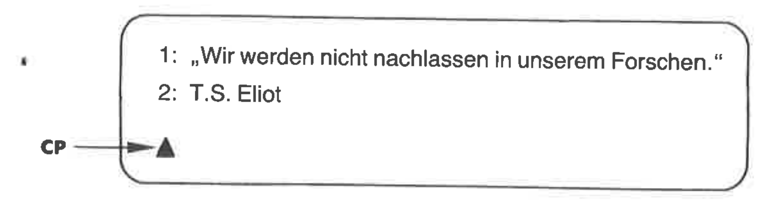


Abb. 4.11: Der CP am Ende des Textpuffers

Im Handbuch des ED-Editors ist angegeben, daß der CP zwischen zwei Zeichen steht. Diese Erläuterung ist jedoch nicht gut umsetzbar; deshalb ist es besser, den CP als unsichtbaren Zeiger zu betrachten, der jeweils

direkt auf den Beginn einer Zeichenkette zeigt. In diesem Buch wird im Gegensatz zu Erläuterungen in Handbüchern von Digital Research angenommen, daß der Zeiger nicht auf eine Position zwischen, sondern auf das rechte der beiden Zeichen (vgl. Abb. 4.12) zeigt.



Abb. 4.12: Verdeutlichung der Position des Zeigers (CP)

Hier wird immer angenommen, daß der CP beim Rücksprung an den Anfang des Textpuffers auf das erste Zeichen zeigt.

ANZEIGE DES TEXTPUFFERINHALTS

Für die Anzeige von Textinhalten im Puffer kann das Format des T-Befehls benutzt werden:

±nT

n kann dabei 0, jede ganze Zahl oder das Symbol # sein (vgl. Abb. 4.13). Wenn n = 0 ist, wird durch den T-Befehl die aktuelle Zeile bis zur Position des CP angezeigt (Zeichen des CP wird nicht mit angezeigt). Die aktuelle Zeile ist immer die Zeile, in der sich auch der CP befindet.

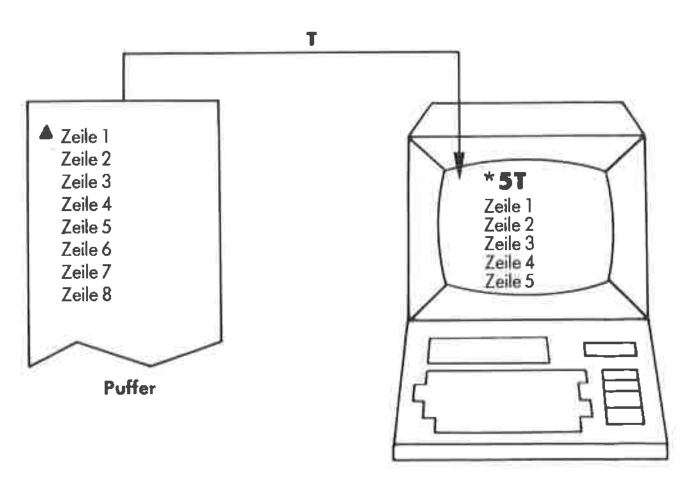


Abb. 4.13: Anzeige von Text

Ohne Angabe einer Zahl wird für n automatisch 1 angenommen. In diesem Fall wird die gesamte aktuelle Zeile angezeigt. Ist n eine positive Zahl, so werden vom T-Befehl, von der aktuellen Zeile beginnend, n Zeilen angezeigt. Ist n eine negative Zahl, so werden n Zeilen vor der aktuellen Zeile angezeigt. In diesem Fall wird die aktuelle Zeile nicht mit angezeigt. Bei der Eingabe des Symbols # nimmt der T-Befehl für n die Zahl 65.535 (maximal erlaubte Anzahl von Textpufferzeilen) an. Hierdurch wird der Textpufferinhalt von der aktuellen Zeile aus gezeigt.

Soll der gesamte Textpufferinhalt angezeigt werden, setzt man den Cursor zuerst an den Anfang des Puffers (mit dem B-Befehl; vgl. auch Abb. 4.14) und gibt danach den T-Befehl mit dem #-Symbol ein.

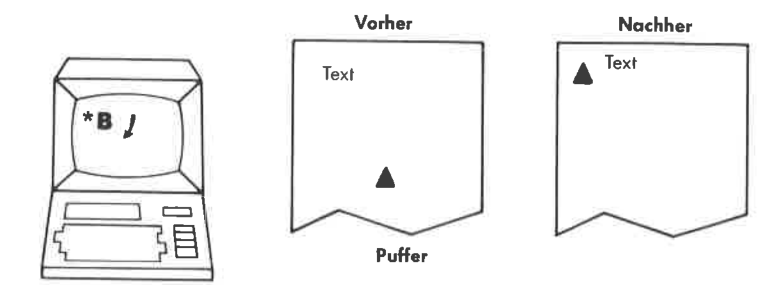


Abb. 4.14: Bewegungen des Cursors

Bei der zusätzlichen Eingabe eines Minuszeichens vor dem #-Symbol (−#) wird der bis zum CP im Textpuffer befindliche Text angezeigt (ausschließlich der aktuellen Zeile).

Beispiel:

```
  *-2T↵
1: "Wir werden nicht nachlassen in unserem Forschen."
2: T.S. Eliot
  :*
```

Wenn vom Bildschirm die Zeilennummern nicht automatisch angezeigt werden (Zahlen mit Doppelpunkt), kann dies durch die Eingabe des V-Befehls erreicht werden:

```
*V↵
*
```

Im obigen Beispiel wird durch : angezeigt, daß der CP am Anfang der Zeile 3 steht. Diese Zeile enthält jedoch noch keinen Text, so daß der CP tatsächlich hinter Zeile 2 (nach dem „Wagenrücklaufzeichen") steht. Der Befehl –2T veranlaßt die ED-Routine zur Anzeige der beiden Zeilen vor dem Zeichenzeiger (steht in der aktuellen Zeile 3).

Es besteht auch die Möglichkeit, durch Eingabe des Doppelpunkts den CP auf eine angegebene Zeile zu setzen. Der Befehl „2:" setzt ihn z.B. zum Anfang der Zeile 2. Soll nur Zeile 1 angezeigt werden, so genügt die Eingabe des Befehls „1:T":

```
2: *1:T↵
1: "Wir werden nicht nachlassen in unserem Forschen."
1:
```

Hierbei wird durch das Argument 1: im T-Befehl der CP gleichzeitig mitbewegt.

Mit einem einfachen T-Befehl wird die aktuelle Zeile angezeigt:

```
2: *T↵
2: T.S. Eliot
2: *
```

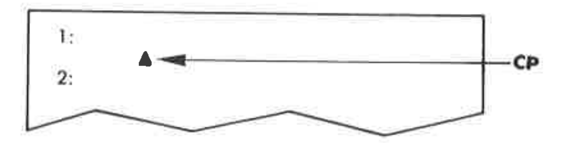
Eine einfache Art, einen gesamten Pufferinhalt anzuzeigen, ist die Kombination des B- und des T-Befehls:

```
1: *B#T↵
1: "Wir werden nicht nachlassen in unserem Forschen."
2: T.S. Eliot
1: *
```

Im obigen Beispiel wird der CP auf das erste Zeichen der ersten Zeile gesetzt. Innerhalb einer Zeile kann der Zeichenzeiger mit dem C-Befehl weiterpositioniert werden:

±nC

Durch den C-Befehl wird der CP um n Positionen nach rechts (nC) bzw. um n Positionen nach links (−nC) geschoben.




```
1: *6C↵
1: *
```

Der Zeichenzeiger wird auf das sechste Zeichen der 1. Zeile (auf e) gesetzt, weil Anführungszeichen auch als einzelne Zeichen interpretiert werden. Der T-Befehl zeigt dann die gesamte Zeile, beginnend vom CP, an (einschließlich des Zeichens, auf das der CP zeigt).

```
1: *T↵
"erden nicht nachlassen in unserem Forschen."
1: *
```

Der Befehl 0T zeigt die Zeile bis zum CP an (ausschließlich des Zeichens, auf das der CP zeigt):

```
1: *0T↵
"Wir w
1: *
```

SICHERN DER DATEI UND BEENDEN DES EDITIERVORGANGS

An dieser Stelle soll erläutert werden, wie der Inhalt des Puffers gesichert und der ED-Editor verlassen werden kann. Der einfachste Weg, den Text zu sichern und den Editor zur gleichen Zeit zu verlassen, ist der E-Befehl:

```
1: *E↵
```

Unabhängig von der augenblicklichen Position des CP im Textpuffer wird der gesamte Puffer (und der restliche Bereich der Quelldatei, der nicht in den Arbeitsspeicher geladen wurde) in die Übergangsdatei geschrieben und diese Ausgabedatei in die Bezeichnung der ursprünglichen Quelldatei umbenannt. Wird mit einer leeren Datei QUOTE.TXT begonnen und der Text des obigen Beispiels hinzugefügt, dann ist nach Abschluß des Editiervorgangs eine Datei mit der Zeile „T.S.Eliot" und eine leere Datei QUOTE.BAK (die Back-up-Datei der ursprünglichen Quelldatei) verfügbar.

LADEN VON DATEIEN IN DEN PUFFER

Existiert bereits eine Textdatei und wird das ED-Programm gestartet, so werden mit dem A-Befehl die Textzeilen in den Puffer geladen. Wird dieser Befehl vergessen, so steht zunächst kein Text im Puffer. Der danach eingefügte Text steht dann am Ende der Quelldatei. Zur Überprüfung des alten Textes kann die gewünschte Anzahl von Textzeilen geladen werden. Das Format dieses Befehls ist:

```
nA
```

Der A-Befehl lädt n Zeilen der Originaldatei (Quelldatei) in den Arbeitsspeicher. Wird n nicht angegeben, so wird nur eine Zeile geladen. Bei der Angabe von # wird die gesamte Quelldatei (maximal 65.535 Zeilen) in den Textpuffer geladen. Weil die meisten Textdateien nicht sehr umfangreich sind, können diese meist ohne Probleme mit #A vollständig geladen werden. Wird für n eine 0 eingegeben, so füllt der A-Befehl die Hälfte des Textpuffers auf (sehr nützlich bei großen Dateien).

In Verbindung mit dem Befehl 0W können dann Teile großer Dateien sehr leicht geladen und gesichert werden (vgl. Kapitel „Schreiben von Textzeilen zur Datei").

Wird nur ein Text der Quelldatei (Original) in den Puffer geladen, so wird beim nächsten A-Befehl an der Stelle mit dem Einladen begonnen, an der mit dem vorangegangenen A-Befehl aufgehört wurde. Es ist dann sehr leicht, z.B. 10 Zeilen zu laden, zu modifizieren, in der Übergangsdatei zu sichern (mit dem W-Befehl) und die nächsten 10 Zeilen (oder mehr) zu laden.

Wenn der Puffer genügend groß ist, können auch wesentlich mehr Textzeichen aus der Quelldatei (eventuell auch die gesamte Quelldatei) in den Puffer geladen werden.

Im folgenden Beispiel wird die Datei QUOTE.TXT als Quelldatei verwendet:

```
A>ED QUOTE.TXT↵
:*A↵           (Laden einer Zeile)
1:*2A↵         (Laden der nächsten zwei Zeilen; in der Datei ist
               jedoch nur eine Zeile gespeichert)
2:*B#T↵        (Sprung an den Anfang und Anzeige des Textpuf-
               ferinhalts)
1: "Wir werden nicht nachlassen in unserem Forschen."
2: T.S. Eliot
1:*
```

Weil in diesem Beispiel nur zwei Zeilen der Datei QUOTE.TXT stehen, können diese auch mit dem Befehl #A geladen werden.

ZEIGER-BEWEGUNGEN IM TEXTPUFFER

Steht der Text im Puffer, so kann dieser an beliebigen Stellen verändert oder neuer Text hinzugefügt werden. Auch können einzelne Textteile gesucht und ausgetauscht oder zusätzlicher Text von einer gesonderten Bibliotheksdatei (als Quelldatei) eingefügt werden. Soll das hier gezeigte Textbeispiel QUOTE.TXT vergrößert werden, so muß zunächst an das Ende der Textdatei gesprungen und dann neuer Text eingegeben werden. Der CP springt an das Ende der Textdatei mit dem Befehl −B:

```
1:*-B↵
 :*
```

Am Ende der letzten Zeile steht ein Wagenrücklaufzeichen (CR = carriage return), das aus den beiden ASCII-Zeichen für RETURN und LINE FEED (Zeilenvorschub) besteht. Der Cursor steht dann am Ende vor einer nicht angezeigten Leerzeile. Aus diesem Grund wird diese Leerzeile so lange nicht angezeigt, bis zusätzliche Zeichen nach Eingabe des I-Befehls eingegeben werden.

Die zusätzlich eingegebenen Zeichen bilden die neue Zeile:

```
1:"Wir werden nicht nachlassen in unserem Forschen."
2: T.S. Eliot
3: Und das Ende unseres Forschens,
4: ist an den Ausgangspunkt zu kommen,
5: und zum erstenmal den Ort zu erkennen."
 :
 ▲
```

Puffer

```
  : *I↵
3: Und das Ende unseres Forschens,↵
4: ist an den Ausgangspunkt zu kommen,↵
5: und zum erstenmal den Ort zu erkennen."↵
6: ^Z↵
  : *
```

Nun kann der gesamte Puffer mit dem B- und dem T-Befehl angezeigt werden:

```
  : *B#T↵
1: "Wir werden nicht nachlassen in unserem Forschen."↵
2: T.S. Eliot
3: Und das Ende unseres Forschens,
4: ist an den Ausgangspunkt zu kommen,
5: und zum erstenmal den Ort zu erkennen."
1: *
```

Die einfachste Möglichkeit, zu einer anderen Zeile zu springen, ist die Verwendung der Zeilennummer. Für die Auswahl einer bestimmten Zeile muß die gewünschte Zeilennummer mit einem folgenden Doppelpunkt eingegeben werden:

```
1: *2:↵
2: *
```

Es ist auch möglich, die letzte Zeile zur Ausführung eines Befehls einzugeben. Hierbei wird der Doppelpunkt vor die Zeilennummer gesetzt. Zur Auswahl eines ganzen Zeilenbereichs muß die erste Zeilennummer, gefolgt von einem Doppelpunkt, und die letzte Zeilennummer mit einem vorangehenden Doppelpunkt eingegeben werden (die Anzeige eines direkt folgenden Zeilenbereichs kann natürlich auch mit dem T-Befehl erreicht werden).

```
1:"Wir werden nicht nachlassen in unserem Forschen."
2: T.S. Eliot
3: Und das Ende unseres Forschens,
4: ist an den Ausgangspunkt zu kommen,
5: und zum erstenmal den Ort zu erkennen."
```

Puffer

```
2: *3::5T↵
3: Und das Ende unseres Forschens,
4: ist an den Ausgangspunkt zu kommen,
5: und zum erstenmal den Ort zu erkennen."
3: *
```

Der Befehl 3::5T entspricht dabei genau 3:3T. Würde stattdessen 5::3T eingegeben werden, so entspräche das dem Befehl 5:−3T.

Wenn eine einzelne Zeile spezifiziert wird, springt der CP an den Beginn dieser Zeile. Wird ein ganzer Zeilenbereich angegeben, springt der CP an den Beginn der ersten Zeile dieses Bereichs, und vom Editor ED wird dann der abgezählte Bereich an den T-Befehl übergeben. Der CP verbleibt jedoch auf der Aufrufstellung (im Beispiel Zeile 3).

Bei der Eingabe des T-Befehls greift dieser auf den Bereichszähler zurück und zeigt den Bereich an. Der CP bleibt jedoch auf Zeile 3.

Genauso kann mit dem L-Befehl innerhalb eines Textes nach oben und unten gesprungen werden. Das benötigte Format ist:

```
#nL
```

Mit dem L-Befehl wird der CP im Textpuffer um +n Zeilen vorwärts oder −n Zeilen rückwärts geschoben. Dabei wird der CP an den Anfang der jeweiligen Zeile gesetzt:

```
3: *1L↵
4: *-2L↵
2: *
```

Bei der Eingabe von 0L (n = 0) springt der CP an den Beginn der aktuellen Zeile. Ausgehend von der aktuellen CP-Position kann auch ein bestimmter Zeilenbereich selektiert werden. Hierzu muß die Nummer der letzten Zeile des Bereichs mit einem vorangestellten Doppelpunkt eingegeben werden:

```
2: *:5T↵
2: T.S. Eliot
3: Und das Ende unseres Forschens,
4: ist an den Ausgangspunkt zu kommen,
5: und zum erstenmal den Ort zu erkennen."
```

AUSTAUSCH, EINFÜGEN UND LÖSCHEN VON TEXT

Textbereiche können ausgetauscht werden, indem mit dem CP an den Beginn des auszutauschenden Textes gesprungen, von dort ab der gewünschte Text gelöscht und der neue Text eingeschoben wird. Genauso kann eine gesamte Zeichenkette aufgesucht und durch eine neue Zeichenkette ersetzt werden. Wird eine Textzeile gelöscht, so werden die Zeilennummern automatisch der neuen Situation angepaßt und geändert:

```
2: *1::5T↵
1: "Wir werden nicht nachlassen in unserem Forschen."
2: T.S. Eliot
3: Und das Ende unseres Forschens,
4: ist an den Ausgangspunkt zu kommen,
5: und zum erstenmal den Ort zu erkennen."
1: 2:↵
2: K↵
2: *1::4T↵
1: "Wir werden nicht nachlassen in unserem Forschen."
2: Und das Ende unseres Forschens,
3: ist an den Ausgangspunkt zu kommen,
4: und zum erstenmal den Ort zu erkennen."
1: *
```

Der K-Befehl (kill) löscht im obigen Beispiel die zweite Zeile und schließt den folgenden Text an. Das Gegenteil wird durch das Einfügen einer Zeile erreicht – die folgenden Zeilen werden nach unten geschoben, um so der neu einzufügenden Zeile Platz zu machen.

Im Format des K-Befehls

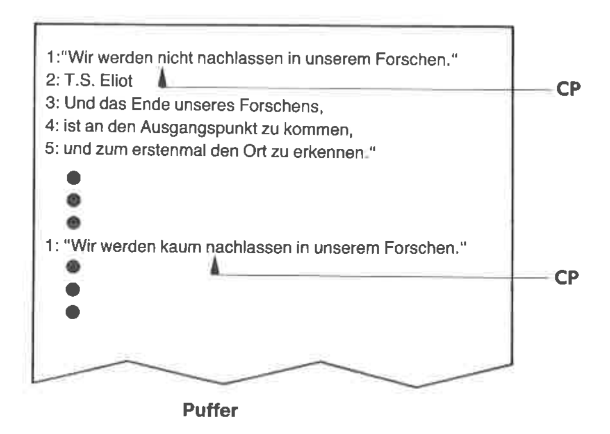
±nK

gibt n die Anzahl der zu löschenden Zeilen an. Wird für n keine Zahl angegeben, so wird die aktuelle Zeile ab der Position des CP gelöscht. Mit nK werden die folgenden n Zeilen bzw. mit −nK die vorhergehenden n Zeilen gelöscht. Wird für n das Symbol # eingegeben, so werden einschließlich der aktuellen Zeile (ab CP-Position) die 65.535 folgenden Zeilen gelöscht (durch −#K werden die vorhergehenden 65.535 Zeilen gelöscht). Hiermit kann ein Text um eine beliebige Anzahl von Textzeilen problemlos bereinigt werden. Die gelöschten Zeilen sind dann auch für nachfolgende Operationen verloren (ausgenommen, sie existieren noch in der Quell- oder Back-Up-Datei). Zum Löschen einzelner Zeichen (nicht Zeilen) wird der D-Befehl verwendet:

±nD

Ist mit dem D-Befehl keine Zahl für n angegeben, so löscht dieser das Zeichen, auf den CP zeigt. Bei der zusätzlichen Angabe einer Zahl werden mit nD die folgenden n Zeichen, ausschließlich des durch den CP bestimmten Zeichens, gelöscht.

Im folgenden Beispiel wird das Wort „nicht" gelöscht und das Wort „kaum" mit dem I-Befehl eingefügt.



Puffer

```
1: *1::4T↵
1: "Wir werden nicht nachlassen in unserem Forschen."
2: Und das Ende unseres Forschens
3: ist an den Ausgangspunkt zu kommen,
4: und zum erstenmal den Ort zu erkennen."
1: *T↵
1: "Wir werden nicht nachlassen in unserem Forschen."
1: *12C↵
1: 5D I kaum^Z0LT↵
1: "Wir werden kaum nachlassen in unserem Forschen."
1: *
```

In diesem Beispiel wird der CP mit dem Befehl 12C auf das dreizehnte Zeichen geschoben. Danach werden 5 Zeichen mit dem D-Befehl gelöscht.

Der anschließend gegebene I-Befehl fügt das Wort „kaum" (mit ^Z begrenzt) vor den CP ein. Mit dem L-Befehl kann der CP an den Beginn der gewünschten Zeilen geschoben, und mit dem T-Befehl können gewünschte Zeilen angezeigt werden.

Eine neue Form des I-Befehls bietet das Format:

Ieinfügung^Z

Der Befehl hat die gleiche Wirkung wie I↵, mit der Ausnahme, daß nicht automatisch auch ein „Carriage Return"-Zeichen für das Generieren einer neuen Zeile eingefügt wird. Wenn eine Gruppe von Zeichen einschließlich eines „Carriage Return"-Zeichens eingefügt werden soll, kann dies mit dem S-Befehl erreicht werden, der im folgenden Abschnitt näher betrachtet werden soll.

AUFSUCHEN UND ERSETZEN VON TEXT

Eine leichte Art, den CP auf eine bestimmte Zeichenfolge im Text zu schieben, bietet der F-Befehl (find = finden). Soll zusätzlich auch diese Zeichenkette durch eine andere ersetzt werden, so kann hierzu der S-Befehl (substitute = ersetzen) verwendet werden.

Zunächst ein Beispiel für den F-Befehl. Es soll das Wort „kaum" in Zeile 1 gefunden werden:

```
1: Fkaum↵
1: *
```

Folgende Abbildung zeigt die aktuelle Stellung des CP nach dem Suchvorgang:

```
".Wir werden kaum ▲ nachlassen in unserem Forschen."
                 CP
```

Der F-Befehl bewegt den CP auf das Zeichen, das direkt dem zuletzt gefundenen Zeichen folgt, wodurch eine Anwendung des D-Befehls sehr erleichtert wird. Danach wird an den Anfang der Zeile zurückgesprungen und der F-Befehl in Kombination mit dem D- und I-Befehl gegeben:

```
1: 0L↵
1: Fkaum^Z-4DInicht^Z0LT↵
1: "Wir werden nicht nachlassen in unserem Forschen."
```

Es werden vier Zeichen links vom CP gelöscht. Der CP steht dann in der Stellung für das Einfügen von „nicht“. Mit dem 0LT-Befehl wird der CP an den Beginn der Zeile geschoben. Wird mit dem F-Befehl nicht noch ein zusätzlicher Befehl eingegeben, kann das gesuchte Wort auch mit einem RETURN abgeschlossen werden.

Dem F-Befehl kann auch eine Zahl vorausgesetzt werden. In diesem Fall wird die n-te gesuchte Zeichengruppe gefunden. Wird beispielsweise der Befehl „5Fdas↵“ eingegeben, so wird der Suchbefehl nach dem fünften Auffinden des Wortes „das“ abgebrochen und der CP auf diese Stelle gesetzt. Mit dem F-Befehl kann nur innerhalb des Textpuffers nach einer bestimmten Zeichenfolge gesucht werden. Die gesamte Quelldatei wird mit dem N-Befehl abgesucht (vgl. auch „Weiterführende ED-Operationen“).

Soll zu einer Gruppe von Zeichen gesprungen werden, in der auch die Zeichenkombination des Carriage Return vorkommt, kann hierzu ein spezieller Steuerbefehl verwendet werden. Dieser Steuerbefehl ist das Zeichen ^L, das die Zeichenfolge RETURN und LINE FEED auffinden kann. Im Beispiel über den „Substitute“-Befehl wird diese Anwendung noch erklärt.

Der S-Befehl (substitute = ersetzen) ist der am häufigsten angewandte Befehl zum Auffinden und Ersetzen von Zeichenketten. Er kombiniert die Einzelbefehle F, D und I. Sein Format lautet:

n S alter text ^Z neuer text { ^Z | ↵ }

Der S-Befehl sucht den gesamten Puffer nach der Zeichenkette „alter text“ ab und ersetzt diese durch die Zeichenfolge „neuer text“. Die neue Zeichenfolge kann sowohl mit ^Z als auch mit RETURN abgeschlossen werden (bei Abschluß mit ^Z können noch weitere Befehle mit angegeben werden). Diese Austauschoperation kann n-mal ausgeführt werden, höchstens jedoch so oft, wie die Zeichenfolge im Textpuffer vorkommt.

Mit dem Befehl „#SFrankfurt^Heidelberg↵“ werden z.B. alle (# steht für 65.535-mal) Zeichenfolgen „Frankfurt“ im Puffer durch „Heidelberg“ ersetzt.

Die einzelnen Sätze des Textbeispiels sollen nun durch einfache S-Befehle in einen Zusammenhang gebracht werden. Zunächst soll das Ende der ersten und der vierten Zeile berichtigt werden:

```
1: *2T↵
1: "Wir werden nicht nachlassen in unserem Forschen."
2: Und das Ende unseres Forschens,
1: *S."^L^Z/^L^ZB4T↵
1: "Wir werden nicht nachlassen in unserem Forschen/
2: Und das Ende unseres Forschens,
3: ist an den Ausgangspunkt zu kommen,
4: und zum erstenmal den Ort zu erkennen/
1: *
```

In der Befehlszeile wird mit dem S-Befehl die Zeichengruppe gesucht, die mit einem Punkt und einem Anführungszeichen beginnt (^L steht für Carriage Return), und durch einen Schrägstrich mit einem Carriage Return ersetzt. Wenn danach direkt der Befehl B4T ausgeführt wird, springt der CP an den Anfang des Textpuffers, worauf die folgenden 4 Zeilen angezeigt werden.

Im folgenden Beispiel werden die Kommata am Ende jeder Zeile durch Schrägstriche (/) ersetzt:

```
1: *2::4T↵
2: Und das Ende unseres Forschens,
3: ist an den Ausgangspunkt zu kommen,
4: und zum erstenmal den Ort zu erkennen/
4: *2:↵
2: *S,^Z/^ZB4T↵
1: "Wir werden nicht nachlassen in unserem Forschen/
2: Und das Ende unseres Forschens/
3: ist an den Ausgangspunkt zu kommen/
4: und zum erstenmal den Ort zu erkennen/
1: *
```

Hier wird der CP an den Anfang der Zeile 2 gesetzt (2:), und die folgenden Kommata werden durch Schrägstriche ersetzt. Danach wird der CP wieder an den Beginn des Puffers positioniert und die folgenden vier Textzeilen angezeigt (B4T).

Jetzt müssen nur noch die beiden Anfangsbuchstaben der dritten und vierten Zeile in Großbuchstaben umgewandelt werden:

```
1: *3:↵
3: *DII^Z1LDIU^ZB4T↵
1: "Wir werden nicht nachlassen in unserem Forschen/
2: Und das Ende unseres Forschens/
3: Ist an den Ausgangspunkt zu kommen/
4: Und zum erstenmal den Ort zu erkennen/
1: *
```

Hierzu wird der CP an den Anfang der dritten Zeile gesetzt, der erste Buchstabe gelöscht (D-Befehl löscht i) und ein großes I eingefügt (I-Befehl). Nach dem Positionieren des CP an den Anfang der nächsten Zeile (1L-Befehl) wird dann wieder der erste Buchstabe gelöscht (D-Befehl löscht u) und ein großes U eingesetzt. Zum Abschluß werden nach dem Positionieren des CP auf den Pufferanfang die vier folgenden Zeilen angezeigt.

SCHREIBEN VON TEXTZEILEN ZUR DATEI

Im allgemeinen wird ein Editiervorgang mit dem E- oder H-Befehl abgeschlossen. Beide Befehle beziehen sich auf den gesamten Pufferinhalt – die Textzeilen werden in die Übergangsdatei geschrieben. Beide Befehle berücksichtigen auch den Rest der Quelldatei (die Zeilen, die noch nicht in den Puffer geladen wurden) und lösen die Umbenennung der betroffenen Dateien aus. Danach steht der modifizierte Text in der neuen Quelldatei. Während der E-Befehl jedoch den Editiervorgang abschließt, wird vom H-Befehl ein Verbleib in der ED-Routine erreicht. Hierdurch ist eine weitere Verarbeitung der Quelldatei ohne erneutes Starten des Editors möglich.

Sollen nur einige Zeilen in die Ausgangsdatei ohne Kopieren des Rests der Quelldatei geschrieben werden, so ist dies mit dem W-Befehl (write) möglich. Er hat das Format

```
nW
```

und schreibt n Zeilen einschließlich der aktuellen Zeile in die Ausgangsdatei. Die Ausgangsdatei ist die Übergangsdatei mit dem Dateityp „$$$". Wird n nicht mit angegeben, so wird nur die aktuelle Zeile in die Ausgangsdatei geschrieben.

Zur Veranschaulichung des W-Befehls sollen einige zusätzliche Textzeilen hinzugefügt und einige Zeilen in die Übergangsdatei geschrieben werden.

```
1: *-BI↵
5: ↵
6:             T.S. Eliot↵
7: ^Z
6: B#T↵
1: „Wir werden nicht nachlassen in unserem Forschen/
2: Und das Ende unseres Forschens/
3: Ist an den Ausgangspunkt zu kommen/
4: Und zum erstenmal den Ort zu erkennen/
5:
6:             T.S. Eliot
1: *3W↵
1: #T↵
1: Und zum erstenmal den Ort zu erkennen/
2:
3:             T.S. Eliot
1: *#W↵
 : *
```

In diesem Beispiel wird zunächst neuer Text an das Ende des Puffers angefügt (−BI-Befehl). Mit RETURN wird eine Leerzeile und mit dem Tabulatorbefehl mehrere Leerzeichen geschrieben. Nach der Positionierung des CP an den Anfang des Puffers wird der gesamte Textpufferinhalt angezeigt (B#T-Befehl).

Durch die weitere Angabe des Befehls 3W werden drei Zeilen, einschließlich der aktuellen Zeile (Zeilen 1, 2 und 3), herausgeschrieben.

Diese Zeilen stehen danach in der Übergangsdatei QUOTE.$$$. Die verbleibenden Textzeilen im Puffer werden dabei neu numeriert. Der letzte Befehl im Beispiel (#W) schiebt die nächsten 65.535 Zeilen (einschließlich der aktuellen Zeile) zur Ausgangsdatei. Der Puffer ist dann leer, und die ED-Routine zeigt nur noch das Symbol :* an.

Nun muß noch der Rest der Quelldatei gesichert (falls sie noch ungeladene Textzeilen enthält) und die alte Quelldatei QUOTE.TXT in QUOTE.BAK umbenannt werden. Mit dem E- und dem H-Befehl wird dies automatisch erreicht.

Eine spezielle Form des Schreibbefehls ist in der Kombination mit dem 0A-Befehl möglich. Mit 0W wird die Hälfte des Puffers beschrieben und mit 0A die Hälfte des Puffers geladen. Hiermit wird der Textpuffer automatisch verwaltet. Er wird deshalb häufig eingesetzt, um z.B. in der Folge eine Pufferhälfte zu laden, diese zu bearbeiten, eine Pufferhälfte wegzuschreiben und eine neue Hälfte zu laden.

Sollen die Zeilen im Textpuffer anders angeordnet oder sollen Textzeilen aus anderen Dateien gemischt werden, gibt man spezielle ED-Befehle ein. Hierbei werden dann einzelne Zeilen in den Puffer geladen und die gewünschte Endversion danach mit dem W-Befehl zur Datei geschrieben (oder mit dem E- bzw. H-Befehl).

WEITERFÜHRENDE ED-OPERATIONEN

Suchvorgänge in der Quelldatei

Mit dem N-Befehl können Textfolgen im Puffer und in der Quelldatei gesucht werden. Er arbeitet auf die gleiche Weise wie der F-Befehl, bricht jedoch nicht am Ende des Puffers mit dem Suchvorgang ab, sondern lädt automatisch weitere Textzeilen, bis die gesuchte Zeichenfolge gefunden ist. Sein Format ergibt sich als

nNtext { ^Z | ↵ }

Wichtig: Die geschweiften Klammern weisen wieder auf die beiden Wahlmöglichkeiten hin. Es wird die n-te mit „text" angegebene Zeichengruppe im Textpuffer gesucht, die nach der Position des CP folgt. Wird die n-te Zeichengruppe nicht gefunden, werden weitere Zeilen von der Quelldatei eingeladen. Die Zeichengruppe kann mit einem ^Z abgeschlossen und danach ein neuer Befehl angefügt werden; anderenfalls ist ein Befehlsabschluß auch durch RETURN möglich. Bei der Anwendung des N-Befehls wird der CP direkt hinter das letzte Zeichen der n-ten gefundenen Zeichengruppe (genau wie beim F-Befehl) gesetzt.

Einfügen von Text aus einer Bibliotheksdatei

Eine „Bibliotheksdatei" ist ein im Handbuch von Digital Research verwendeter Ausdruck für jede Datei vom Dateityp „LIB" (z.B. BEISPIEL. LIB). Eine Bibliotheksquelldatei kann als Quelldatei für das Einladen häufig verwendbarer Textzeilen in den Pufferbereich anderer Quelldateien verwendet werden.

Hierzu muß zunächst eine gefüllte Textdatei vom Dateityp „LIB" zur Verfügung stehen. Im Format

Rdateiname

wird vom R-Befehl die durch den Dateinamen „dateiname" angesprochene Datei vom Typ „LIB" aufgesucht und vor der aktuellen CP-Position des Textpuffers eingefügt. Hierbei wird die gesamte LIB-Datei bis zum Erkennen eines EOF-Zeichens (^Z) geladen und eingefügt.

Übertragen und Einfügen von Textzeilen aus der Übergangsdatei

Mit dem X-Befehl können Textzeilen vom Puffer in eine zweite Übergangsdatei und von dieser Textzeilen wieder in den Puffer übertragen werden. Die zweite Übertragungsdatei ist mit X$$$$$$$.LIB bezeichnet und existiert nur so lange, bis der Editierbetrieb abgebrochen wird.

Jeder erlaubte Abbruch im Editor löscht diese Datei. Wird der Editor über ^C (Warmstart) verlassen, bleibt die Datei zurück und wird erst nach einem erneuten Start des Editors gelöscht.

Das Format des X-Befehls lautet:

```
nX
```

wobei mit dem X-Befehl n Zeilen (ausgehend von der aktuellen Zeile) in die Übergangsdatei X$$$$$$$.LIB geschrieben werden.

Nach dem Übertragungsvorgang können die unerwünschten Zeilen mit dem K-Befehl gelöscht werden. Innerhalb der Übergangsdatei sind die übertragenen Zeilen in der gleichen Folge der nachfolgenden X-Befehle gespeichert. Jeder Befehl erzeugt neuen Text in aufeinanderfolgenden Blöcken.

Die erstellten Textzeilen können wieder mit dem R-Befehl (ohne Angabe des Dateinamens) geladen werden. Die übertragenen Textzeilen stehen dann vor dem CP (ähnlich dem I-Befehl). Hier wird im Gegensatz zum oben erläuterten R-Befehl nicht der gesamte Inhalt der Bibliotheksdatei geleert, sondern nur in den Puffer kopiert. Er kann somit mehrfach geladen werden (sehr nützlich bei immer wiederkehrenden Textzeilen).

Entleert (d.h. gelöscht) wird die Übergangsdatei durch die Ausführung eines Befehls im Format 0X (n = 0). Soll die Bibliotheksdatei X$$$$$$$.LIB weiter verwendet werden, so empfiehlt sich ein Verlassen des Editors über ^C und die sofortige Umbenennung in eine andere LIB-Datei. Nur so wird diese LIB-Datei beim nächsten Aufruf des Editors nicht gelöscht.

Juxtaposition

Mit dem J-Befehl können drei Zeichengruppen innerhalb eines Textes zusammengeschoben werden. Er sucht eine erste Textgruppe auf, fügt eine zweite Textgruppe an und löscht alle Zeichen bis zum Erreichen der dritten Textgruppe. Somit sind danach alle drei Textgruppen nebeneinander gestellt.

Das Format des J-Befehls lautet:

```
nJZeichenfolge1^ZZeichenfolge2^ZZeichenfolge3 { ^Z }
                                              { ↵  }
```

Nach Eingabe des J-Befehls beginnt der Editor, ausgehend vom CP, mit der Suche nach der ersten Zeichenfolge. Ist diese gefunden, so wird die zweite Zeichenfolge direkt angeschlossen und der CP an das Ende der zweiten Zeichenfolge gesetzt. Der nachfolgende Löschvorgang löscht alle Zeichen zwischen Zeichenfolge 2 und Zeichenfolge 3 und setzt den CP auf das erste Zeichen der Zeichenfolge 3.

Wird die Zeichenfolge 3 nicht gefunden, so findet auch kein Löschvorgang statt. Dieser J-Vorgang wird insgesamt n-mal oder bis zum Erreichen des Pufferendes durchgeführt.

Eine Anwendung des J-Befehls ist das definierte Löschen von Textbereichen. Hierbei wird als Zeichenfolge 1 meist das Ende der Zeile von einem zu löschenden Textbereich, als Zeichenfolge 2 ein Leerzeichen und als Zeichenfolge 3 eine Wagenrücklauf-Sequenz (^L) angegeben.

Wiederholung eines Befehlssatzes

Mehrere ED-Befehle können zu einem einzigen Befehl zusammengefügt und gespeichert werden. Dieser hierzu einsetzbare M-Befehl kann beliebig oft wiederholt werden. Im folgenden Format des M-Befehls:

```
                  { ^Z }
nMBefehlsfolge    { ↵  }
```

wird eine Gruppe von Befehlen, eine Befehlsfolge, der M vorangestellt ist, n-mal ausgeführt. Wird für n 0, 1 oder nichts angegeben, so wird der Makrobefehl so oft ausgeführt, bis ein Fehler erkannt wird (z.B. Erreichen des Pufferendes).

Ein Beispiel ist der vollständige Austausch der Buchstabengruppe „Frankfurt“ durch „Heidelberg“ und das Anzeigen jeder veränderten Zeile:

```
MSFrankfurt^ZHeidelberg^Z0TT
```

ED-FEHLERMELDUNGEN

Treten Fehler auf, gibt der Editor die Meldung „BREAK x AT c“ aus, wobei x eines der Fehlersymbole und c den ausgeführten ED-Befehl angibt. Folgende Fehler können auftreten:

Symbol	Bedeutung
#	Such-Fehler. ED kann die im F-, S- oder N-Befehl angegebene Zeichenfolge nicht finden.
?c	Falsches Befehlszeichen c. ED erlaubt keine weiteren Befehle zusammen mit dem E-, H-, Q- oder O-Befehl.
0	.LIB-Datei nicht gefunden.
>	Puffer voll.
E	Befehl abgebrochen.
F	Fehler in der Dateiverwaltung (Diskette oder Directory voll).

Abb. 4.15: ED-Fehlermeldungen

Darüber hinaus können gleich nach dem Start des Editors folgende Fehlermeldungen auftreten:

```
** FILE IS READ ONLY **
SYSTEM FILE NOT ACCESSIBLE
```

In diesen Fällen müssen Schreibschutz- und System-Attribut zurückgesetzt werden (siehe STAT-Befehl in Kapitel 2).

Treten Fehlermeldungen des Systems auf (z.B. „DISK READ ONLY“ oder „SELECT“, führt das zum Abbruch des Editors ohne Rettung des Zeichenpuffers.

ZUSAMMENFASSUNG

In diesem Kapitel wurde die Anwendung des CP/M-Editors besprochen. Der ED-Editor ist ein allgemein einsetzbarer Editor, mit dem Texte in Dateien mit wenigen Steuerbefehlen modifiziert werden können. Genauso häufig wird die ED-Routine zur einfachen Generierung von Dateien eingesetzt. Obwohl dieser Editor über weit weniger Funktionen als gebräuchliche Textverarbeitungsprogramme verfügt, kann er für das Schreiben von Briefen und Dokumenten eingesetzt werden.

Der Editor kann darüber hinaus als leicht einsetzbares Hilfsmittel zur Korrektur und Modifikation existierender Dateien verwendet werden (z.B. auch für die Bereinigung teilweise zerstörter Dateien).

So lassen sich auf einfache Weise neue Wörter oder Zeilen in vorhandene Dateien einfügen. Eine Zusammenfassung aller ED-Befehle und Steuerzeichen befindet sich in Anhang D und E.

Über den einfachen Umgang mit dem ED-Editor hinaus wurden spezielle Steuerbefehle und Anwendungen behandelt; es wurden Konzepte für eine effektive Dateiorganisation dargestellt und die Übertragungsprozeduren im groben Ablauf zwischen Diskettenlaufwerk, Speicher und Bildschirmterminal analysiert.

Hierdurch wird auch der wesentliche Teil anderer Texterstellungs-Programme für Mikrocomputersysteme vom Konzept her erfaßt und für weitere Analysen leicht zugänglich.

Kapitel 5

Struktur von Betriebssystemen der CP/M-Familie

EINFÜHRUNG

In diesem Kapitel werden die Operationen, die innerhalb eines Betriebssystems der CP/M-Familie ablaufen, näher beschrieben. Dieses Kapitel ist besonders für Anwender interessant, die sich für Strukturen und Abläufe in Betriebssystemen interessieren oder einige der CP/M-Routinen ändern wollen. Sollen jedoch lediglich die schon beschriebenen Funktionen in der bereits dargestellten Art genutzt werden, so sind diese Kenntnisse nicht von besonderer Bedeutung.

Sehr wertvoll ist dieses Kapitel für Systemprogrammierer oder erfahrene CP/M-Anwender, die zusätzlich noch die internen Arbeitsschritte verstehen wollen. Für diese Leser, und weil ein Buch über CP/M auch dessen innere Struktur beschreiben muß, wurde dieses Kapitel geschrieben.

Zunächst wird eine Übersicht über die CP/M-Operationen gegeben. Hierbei werden die verwendeten logischen Module, deren Funktionen und deren Zusammenwirken betrachtet. Nach der Analyse der „Memory Allocation" (Speicherzuordnung), mit der die einzelnen Softwaremodule im internen Arbeitsspeicher definiert sind, wird das System der Dateiorganisation betrachtet. Wir gehen dabei von CP/M-80 (Version 2.2) aus, dessen Dateienverwaltung auch bei den Nachfolgesystemen (MP/M, CCP/M, CP/M-Plus) beibehalten wurde.

Danach werden die drei Softwaremodule mit den entsprechenden Befehlen im Detail vorgestellt. In einem anderen Abschnitt werden die Probleme bei der CP/M-Adaption neuer Hardwarekonfigurationen kurz behandelt und ein Beispiel zur Änderung des CP/M-Systems in ein menügesteuertes System diskutiert. Abschließend werden in gesonderten Abschnitten die speziellen Eigenschaften von CP/M-Plus, MP/M und CCP/M erläutert.

Allgemeine Arbeitsweise eines Plattenbetriebssystems

Das Betriebssystem eines Computers hat die Aufgabe, dem Benutzer bzw. den Benutzerprogrammen die Handhabung physikalischer Schnittstellen abzunehmen und eine standardisierte Programmumgebung zu

schaffen. Ein Plattenbetriebssystem (Disk Operating System, DOS) zeichnet sich besonders dadurch aus, daß es als externes Speichermedium Magnetplattenlaufwerke verwaltet.

STRUKTUR DER CP/M-BETRIEBSSYSTEME

CP/M (und seine Nachfolger) kann sowohl logisch als auch physikalisch in verschiedene Funktionsblöcke eingeteilt werden. Wir fangen bei ihrer Beschreibung auf der physikalischen Ebene an. Zur Kommunikation mit den Peripheriegeräten und zur speziellen Anpassung des Systems auf die gegebene Hardware dient das Modul BIOS (Basic I/O System) bzw. bei MP/M das XIOS (eXtended I/O System). Das BIOS ist für jeden Rechnertyp anders; der Rechnerhersteller (oder auch der Benutzer) kann es für sich „maßschneidern".

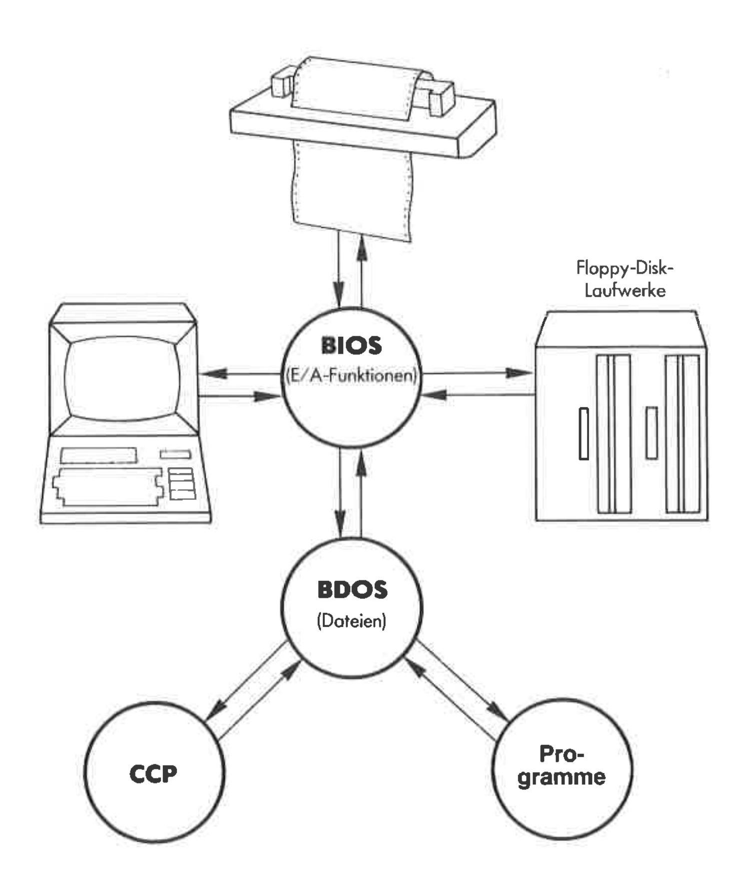


Abb. 5.1: CP/M-Struktur

Die eigentlichen Verwaltungsaufgaben übernimmt das Modul BDOS (Basic Disk Operating System), das wiederum auf das BIOS zugreifen kann. Über das BDOS finden in der Regel alle Kommunikationen zur Außenwelt und zu den Plattenspeichern statt. Das BDOS ordnet Dateien, die über einen Dateinamen angesprochen werden können, Speicherplatz auf den Magnetplatten zu. Eine Datei entspricht, wenn man die Pascal-Terminologie zugrunde legt, einem FILE OF RECORDS mit einer festen Recordlänge (physikalische Satzlänge) von 128 Bytes.

Der dritte wichtige Teil des Betriebssystems ist der Befehls-Interpreter (bei CP/M: Console Command Processor CCP, bei MP/M: Command Line Interpreter CLI). Er hat die Aufgabe, den ersten Kontakt zwischen Benutzer und Rechner herzustellen sowie das Ausführen von Befehlen und das Starten von Programmen zu veranlassen. Der Befehlsinterpreter kann als eine Art „Ur-Programm" verstanden werden. Er greift zur Ausführung seiner Funktionen auf das BDOS zu. Abb. 5.1 zeigt den Zusammenhang zwischen BIOS, BDOS und CCP.

SPEICHERAUFTEILUNG

Das Prinzip der Speicheraufteilung bei der CP/M-Familie sieht so aus, daß das Betriebssystem am oberen Ende des Speichers steht und das Benutzerprogramm den unteren Bereich bis zum Beginn des Systems zur Verfügung hat. Die untersten 256 Bytes sind reserviert für die Kommuni-

Hexa	Dezimal	
FFFFH	65535	
		frei
C000H	49152	
		BIOS
B200H	45568	
		BDOS
AC00H	44032	
		CCP
A400H	41984	
		TPA
0100H	256	
		Base-Page
0000H	0	

Abb. 5.2: Speicheraufteilung bei einem CP/M-80-System mit 48 Kbyte

(Anm.: Der CCP kann während einer Programmausführung überlagert werden.)

kation zwischen Benutzerprogramm und System (bei den 86er-Systemen beziehen sich diese Angaben jeweils auf die Basis eines Speichersegments). Der für den Benutzer freie Speicherplatz wird auch als TPA (Transient Program Area) bezeichnet. Abb. 5.2 zeigt die Speicheraufteilung eines CP/M-80-Systems mit 48 KByte RAM.

CP/M setzt voraus, daß der Speicherraum von 0000 anfangend durchgehend mit RAM (Schreib/Lese-Speicher) bestückt ist. (Für einige Rechner, die Festwertspeicher im unteren Adreßbereich haben, gibt es Spezialversionen.) Der minimale Gesamt-Speicherbedarf bei CP/M-80 ist 20 KByte.

Die unterste Seite (für MP/M gilt das für jedes Speichersegment) wird als Base-Page bezeichnet. Eine Seite (Page) ist jeweils 256 Bytes (100 Hex) lang. Diese Base-Page dient vor allem zur Kommunikation zwischen Anwenderprogramm und System. Die Aufteilung ist in Abb.5.3 zu sehen.

Hexa	Dezimal	
0100H	256	Befehls-Zeile
0080H	128	
007CH	124	FCB 2
005CH	92	FCB 1
0005H	5	JMP BDOS
0000H	0	JMP WSTART

Abb. 5.3: Aufteilung der Base-Page

DAS „FILE SYSTEM" (DATEIVERWALTUNG)

Eine der wesentlichen Funktionen eines plattenorientierten Betriebssystems (Disk Operating System = DOS) ist die Realisierung einer effektiven Dateiverwaltung auf der Platte bzw. Diskette. Grundlage hierzu bildet im CP/M das BDOS. Detailkenntnisse über dieses CP/M-Modul sind deshalb unerläßlich.

Eine Datei (file) ist eine logische Einheit, die Texte, Daten oder Programme enthalten kann. Das plattenorientierte Betriebssystem hat die

Aufgabe, eine logische Verbindung zum physikalischen Ort der Speicherung auf den externen Speichermedien (hier Diskette) herzustellen. Dies soll im folgenden erläutert werden.

Wie bereits ausgeführt, ist eine Diskette nach Spuren und Sektoren organisiert. Jeder Sektor auf einer Standard-8-Zoll-Single-Density-Diskette enthält 128 Bytes an Informationen. Nach der CP/M-Terminologie wird diese Einheit „Satz" (record) genannt. Jede Datei auf einer Diskette besteht aus einer bestimmten Anzahl von Records. Da die Dateien auf der Diskette in unvorhersehbarer Reihenfolge angelegt, beschrieben und gelöscht werden, kann nicht garantiert werden, daß alle Records einer Datei physikalisch hintereinander abgelegt werden. Es wird deshalb eine Datenstruktur (Belegungsplan der betreffenden Platte) angelegt, die Auskunft über die genaue Belegung der Diskette gibt. Bei der CP/M-Familie ist das folgendermaßen gelöst:

Der gesamte Speicherplatz der Diskette (abzüglich der Systemspuren) wird in Blöcke unterteilt, deren Größe von der Gesamtkapazität abhängt. Bei Standard-Floppies sind sie 1 KByte lang, entsprechen also 8 Records. Innerhalb eines Blocks liegen die Records einer Datei sequentiell hintereinander (mit der Einschränkung des „Skewing", siehe Abschnitt „Versetzte Sektorfolge"). Diese Blöcke werden nun an die Dateien vergeben, wobei jeder Block nur einmal vergeben werden kann, auch wenn er nur teilweise belegt ist. Ein Belegungsplan dieser Blöcke befindet sich in der Directory (Inhaltsverzeichnis), die selbst die ersten Blöcke einnimmt. Dadurch, daß statt der einzelnen Records ganze Record-Blöcke vergeben werden, verringert sich die Größe des Belegungplans in der Directory. Die Wahl der Blockgröße stellt einen Kompromiß zwischen vielen Belegungs-Einträgen und großem Speicherraum-Verschnitt (bei großen Blökken und vielen Dateien) dar. Jeder Directory-Eintrag (Directory Entry) beinhaltet den Dateinamen und die Verwaltung von bis zu 16 Blöcken. Die Blöcke sind numeriert von 0 bis 255 (bei großen Festplattenlaufwerken können es auch 8 Blöcke mit Nummern von 0 bis 65.535 sein). Damit kann jeder Eintrag einer Standard-Floppy 16 KByte Dateilänge verwalten. Ist die Datei länger, so werden weitere Einträge verwendet.

Ein Directory-Eintrag besteht aus 32 Bytes und ist im Directory-Bereich auf der Diskette abgelegt (vgl. Abb. 5.4). Immer wenn eine Datei aktiv ist, wird der entsprechende Eintrag in die TPA gebracht, von wo aus durch das Betriebssystem ein schneller Zugriff ermöglicht wird.

Alle Dateien sind demnach über Verwaltungseinträge logisch den physikalischen Sektoren auf der Diskette zugeordnet, so daß von der Directory aus eine schnelle Identifikation möglich ist.

Ein Grunderfordernis jedes guten Betriebssystems ist die Möglichkeit des Dateizugriffs über selbst gegebene symbolische Namen (symbolic names). Im CP/M wird diese Möglichkeit, wie auch die zusätzliche

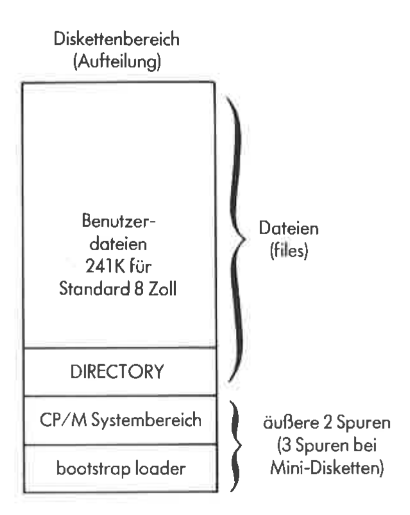


Abb. 5.4: Diskettenbereiche in CP/M

Anwendung von Dateitypen (type of file), unterstützt. Deshalb müssen vom System Routinen bereitgestellt werden, die aus einer logischen Dateibezeichnung den physikalischen Ort der Datei ableiten. Dies geschieht zunächst durch das Absuchen der Directory nach der gegebenen Dateibezeichnung. Die Verbindung zwischen der Dateibezeichnung und der eigentlichen Datei wird durch den sogenannten File Control Block (FCB) hergestellt. Der FCB entspricht in seinem Aufbau weitgehend den Directory-Einträgen, was z.B. ein schnelles Auffinden des entsprechenden Eintrags ermöglicht. Dateiname und Dateityp sind hintereinander gespeichert, so daß sich folgendes Schema ergibt (Abb. 5.5):

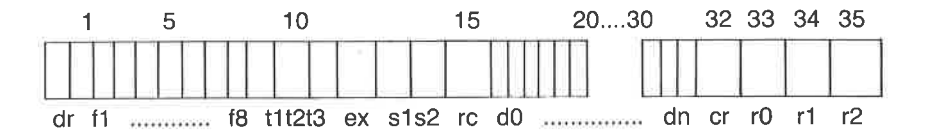


Abb. 5.5: File Control Block (FCB)

Eine weitere wichtige Aufgabe eines leistungsfähigen Datei-Verwaltungssystems ist die Vergabe von Kennungen zur Dateisicherung. So können Dateien z.B. als Read/Only (Nur-Lese)-Dateien oder Read/Write (Schreib/Lese)-Dateien festgelegt werden. Eine bestimmte Anzahl von Bytes ist für die Verwaltung des Systems reserviert.

Im einzelnen ergeben sich folgende Bedeutungen:

dr (Pos. 0) — Das erste Byte dient als Laufwerksauswahlcode (drive code). Bei der Speicherung des FCB als Directory-Eintrag dient es zur Kennzeichnung des Benutzer-Bereichs (user code).

f1…f8 — File Name. Dies ist der Dateiname, der bis zu acht Zeichen enthält. Jedes vom Benutzer nicht angegebene Zeichen wird als ASCII-Leerzeichen interpretiert.

t1…t3 — File Type. Dies ist der Dateityp bzw. die Dateikennung; sie hat bis zu drei alphanumerische Zeichen und wird bei Nichtangabe mit ASCII-Leerzeichen aufgefüllt. Immer wenn eine Directory ausgegeben wird, werden Dateiname und Dateityp angegeben. Die jeweils obersten Bits dienen zur Kennzeichung von Attributen.

ex (Pos. 12) — Extent. Nummer des Directory-Eintrags für diese Datei, beginnend mit 0.

s1 und s2 — Für den BDOS-internen Gebrauch.

rc (Pos. 15) — Record Counter (Satz-Nummer). Die Anzahl der Records in den letzten von diesem FCB verwalteten 16 KByte (0…127).

d0…dn — Belegte Blocknummern für den Directory-Eintrag, verwaltet vom BDOS.

cr (Pos. 32) — Current Record. Position eines Record-Zeigers innerhalb eines 16 KByte-Bereichs bei Lese- und Schreiboperationen. Keine Speicherung in der Directory.

r0,r1,r2 — Random Record. Absolute Record-Position bei „Random"-Lese- und Schreiboperationen. Keine Speicherung in der Directory.

ABLAUF NACH EINGABE EINES BEFEHLS

Nach dem Kaltstart sowie jedem Warmstart wird der CCP als erstes Programm ausgeführt. Dieser gibt seine Bereitschaftsmeldung auf die Konsole aus (das selektierte Laufwerk und das Zeichen >) und wartet auf eine Eingabezeile. Nachdem ihm eine Eingabezeile vom BDOS übergeben worden ist, vergleicht er das erste Wort mit den Bezeichnungen seiner

eingebauten Befehle. Bei Übereinstimmung führt er den Befehl aus und beginnt wieder mit der Ausgabe des Bereitschaftszeichens. Ist das Wort nicht in seinem Befehlsvorrat, so interpretiert er es als eine Befehlsdatei. Er lädt diese Datei (über BDOS) in den Speicherbereich ab 100H. Der Rest der Eingabezeile wird, in Großbuchstaben umgewandelt, zur Adresse 80H der Base-Page kopiert. Bis zu 2 Argumente werden zudem noch als File Control Blocks zu den Adressen 5CH und 7CH kopiert. Dann beginnt die Programmausführung mit einen Sprung zur Adresse 100H. Das Programm wird beendet durch einen Warmstart (Sprung zur Adresse 0 oder BDOS-Funktion 0) oder einen Rücksprung zum CCP (nur mit Vorsicht zu verwenden).

BDOS-OPERATIONEN

Ein vom CCP geladenes Programm steht in dem Speicherbereich, der als TPA bezeichnet wird. Es kann auf die Funktionen des Systems direkt über einen Unterprogramm-Aufruf (CALL) zur Adresse 05H zurückgreifen. Diese Adresse ist die Verzweigungsadresse in das Betriebssystem. Dieser feste Ansprungspunkt ist unabhängig von der jeweiligen Speichergröße und übergibt die Steuerung an CP/M. Die aktuellen CP/M-Routinen sind im oberen Bereich des Speichers resident (siehe Abb. 5.2). Zu einem Betriebssystemaufruf muß noch ein zusätzlicher Parameter, entsprechend der gewünschten CP/M-Routine, angegeben werden. Dieser Parameter entspricht einer Funktionsnummer und muß in Register C des Mikroprozessors 8080 bzw. Z80 geladen werden. Beim CP/M 2.2 stehen 39 BDOS-Funktionen zur Verfügung, bei CP/M-Plus und MP/M sind es erheblich mehr.

Auf alle Funktionen des BDOS kann über den einzigen Eintrittspunkt auf Adresse 0005H zugegriffen werden. Die Funktion wird durch die Angabe einer Funktionsnummer selektiert. Zusätzlich können noch Parameter (z.B. die Adresse eines File Control Blocks) übergeben werden. Soll z.B. das ASCII-Zeichen B zum Terminal gesendet werden, muß die Funktionsnummer der Funktion „console output" (Funktion 2) im C-Register und der ASCII-Wert von B im Register E gespeichert werden.

Die Anzahl der von Digital Research implementierten Funktionen hat sich mit der Entwicklung neuerer Versionen ständig erhöht. Es ist jedoch stets auf größtmögliche Verträglichkeit mit den Vorgänger-Versionen geachtet worden. Die folgenden Systemfunktionen der CP/M-80-Version 2.2 sind bis auf Funktion 3,4,7 und 8 bei allen weiteren Versionen beibehalten worden, doch sollte im Einzelfall die Dokumentation von Digital Research bezüglich Erweiterungen und Änderungen konsultiert werden:

0 System Reset. Ein Warmstart wird ausgeführt und damit das System in den Anfangsstatus zurückgesetzt. Es erfolgt kein Rücksprung zum aufrufenden Programm.

1 Console Input. Ein Zeichen wird von der Konsole eingelesen, als sogenanntes Echo an die Konsole zurückgesendet und dann dem aufrufenden Programm übergeben.

2 Console Output. Ein Zeichen wird auf die Konsole ausgegeben. Tabulatoren werden berücksichtigt und die Eingabe auf ^P und ^S überprüft.

3 Unterschiedliche Bedeutung bei den verschiedenen CP/M-Varianten.

4 Unterschiedliche Bedeutung bei den verschiedenen CP/M-Varianten.

5 List Output. Ausgabe eines Zeichens auf den Drucker.

6 Direct Console I/O. Direktes Ansprechen der Eingangs-/Ausgangstreiber für die Konsole ohne die Zusatzfunktionen von 1 und 2.

7 Unterschiedliche Bedeutung bei den verschiedenen CP/M-Varianten.

8 Unterschiedliche Bedeutung bei den verschiedenen CP/M-Varianten.

9 Print String. Ein ganze Zeichenkette wird bis zur Erkennung eines Ende-Zeichens zur Konsole gesendet.

10 Read Console Buffer. Eine Eingabezeile wird von der Konsole gelesen und dem aufrufenden Programm übergeben.

11 Get Console Status. Überprüfung auf erfolgte Eingabe von der Konsole.

12 Return Version Number. Diese Funktion dient zur Feststellung der Version des Systems.

13 Reset Disk System. Alle Laufwerke werden „ausgeloggt“ und dann Laufwerk A angewählt. Kann z.B. zum Wechsel von Disketten während des Programmlaufs dienen.

14 Select Disk. Ein bestimmtes Laufwerk wird angewählt.

15 Open File. Eine Datei wird eröffnet. Dem BDOS wird dazu (wie bei den folgenden Datei-Operationen auch) die Adresse eines File Control Blocks übergeben.

16 Close File. Abschluß der Arbeit mit einer Datei. Nach Schreiboperationen wird erst jetzt die korrekte Verwaltungsinformation in der Directory gesichert.

17 Search for first. Suche nach einer Datei ab Anfang der Directory. Mit dieser Operation können auch durch Benutzung von Fragezeichen im FCB mehrere Dateien gleichzeitig gesucht werden.

18 Search for next. Suche nach weiteren Dateien nach Aufruf von Funktion 17.

19 Delete File. Die angegebene Datei wird gelöscht (wenn nicht schreibgeschützt).

20 Read Sequential. Der nächste Record der Datei wird gelesen und das rc-Feld des FCB weitergezählt.

21 Write Sequential. Der nächste Record der Datei wird geschrieben und das rc-Feld des FCB weitergezählt.

22 Make File. Eine neue Datei wird angelegt. Es erfolgt zunächst ein Leereintrag in der Directory, da die Datei ja noch keinen Inhalt hat.

23 Rename File. Umbenennen einer Datei. Diese Funktion erfordert einen speziellen FCB, der sowohl den alten wie auch den neuen Namen enthält.

24 Return Login Vector. Dem aufrufenden Programm wird eine Bit-Liste aller „eingeloggten“ Laufwerke übergeben.

25 Return Current Disk. Die Nummer des angewählten Laufwerks wird dem aufrufenden Programm übergeben.

26 Set DMA Address. Die Speicher-Adresse für den Transfer mit den folgenden Datei-Lese- und -Schreiboperationen wird gesetzt.

27 Get Allocation Vector Address. Die Adresse einer Bit-Tabelle, die die belegten Blöcke auf dem selektierten Laufwerk darstellt, wird dem aufrufenden Programm übergeben (nützlich z.B. zur Feststellung des freien Speicherplatzes einer Diskette).

28 Write Protect Disk. Der Schreibschutz für das selektierte Laufwerk wird eingeschaltet.

29 Get Read-Only Vector. Dem aufrufenden Programm wird eine Bit-Liste aller schreibgeschützten Laufwerke übergeben.

30 Set File Attributes. Die Datei-Attribute (wie System oder R/O) lassen sich mit dieser Funktion setzen und zurücksetzen.

31 Get DPB Address. Die Adresse des Disk Parameter Blocks, eine im BIOS stehende Laufwerks-Parameter-Tabelle, wird dem aufrufenden Programm übergeben.

32 Set/Get User Code. Der augenblicklich aktive Benutzerbereich kann hiermit gesetzt oder abgefragt werden.

33 Read Random. Lesen eines frei wählbaren Records aus der Datei.

34 Write Random. Schreiben eines frei wählbaren Records.

35 Compute File Size. Die Anzahl der Records dieser Datei wird ermittelt.

36 Set Random Record. Die absolute Record-Nummer der augenblicklichen Position in der Datei wird im FCB abgelegt.

37 Reset Drive. Ein oder mehrere Laufwerke werden „ausgeloggt“.

38 Access Drive. Vorbelegung von Laufwerken bei Multitasking-Systemen.

39 Free Drive. Funktion zur Freigabe von Laufwerken bei Multitasking-Systemen.

40 Write Random With Zero Fill. Wie Funktion 34, doch zuvor Beschreiben des ganzen Blocks mit Nullen.

VERSETZTE SEKTORFOLGE (Interleave, Skewing)

Die Speicherung von Dateien in hintereinander liegenden Sektoren auf der Diskette kann u.U. zu langen Zugriffszeiten führen. Sie entstehen dadurch, daß das System zeitlich nicht in der Lage ist, die Sektoren direkt hintereinander zu lesen oder zu schreiben. Es muß somit jedesmal eine ganze Plattenumdrehung abwarten. Abhilfe schafft hier eine Umsortierung der Sektor-Reihenfolge: Skewing. Zu unterscheiden ist dabei das Software-Skewing, das durch eine Zuordnungstabelle im BIOS gemacht wird, und das Hardware-Skewing, das dadurch zustande kommt, daß softsektorierte Disketten beim Formatieren eine nicht sequentielle Sektorfolge erhalten. Standard bei CP/M sind 8-Zoll-Single-Density-Disketten mit einem Software-Skewfaktor von 6. Das heißt, die Sektoren werden in der Reihenfolge 1-7-13-19 usw. angesprochen.

BLOCKING – DEBLOCKING

Die Dateien der CP/M-Familie werden in 128-Byte-Einheiten, den sogenannten Records, abgespeichert. Bei Single-Density Standard-Disketten entspricht das genau der Länge eines Sektors. Inzwischen werden jedoch vielfach Plattenspeicher mit höherer Aufzeichnungsdichte eingesetzt, wobei auch die Sektorlänge größer ist. Üblich sind dabei Sektorlängen von 256, 512 oder 1024 Bytes. Die Platten-Controller können also nur ganze Sektoren lesen oder schreiben. Da die Programme jedoch mit 128-Byte-Einheiten arbeiten, müssen diese physikalischen Sektoren in logische Records unterteilt werden. Bei CP/M-Plus wird das vom BDOS getan; bei den anderen Systemen müssen spezielle Algorithmen im BIOS (bzw. XIOS) dieses sogenannte „Deblocking“ übernehmen.

SPEICHERAUFTEILUNG BEI CP/M-PLUS

CP/M-Plus wird in zwei Versionen angeboten: Eine „gebankte“ und eine „ungebankte“ Version, die in ihren Möglichkeiten etwas eingeschränkt ist. Der Begriff „Banking“ bezieht sich auf eine spezielle Art der Speicherverwaltung. Sie wird erforderlich, wenn der anzusprechende Speicherraum größer ist als der, den der Prozessor direkt adressieren kann. Von den Mikroprozessoren der 80er Familie können direkt nur 64 KByte adressiert werden. Nimmt das Betriebssystem selbst viel Platz ein (was bei CP/M-Plus der Fall ist), bleibt für die Anwenderprogramme nur noch wenig Speicher übrig. Aus diesem Grunde verlagert man beim gebankten System den größten Teil des Betriebssystems in einen anderen 64 KByte-Speicherraum, eine sogenannte Bank. Die Rechner-Hardware muß natürlich entsprechend vorbereitet sein, um zwischen Speicherbanken umschalten zu können (z.B. mit einer Memory Management Unit, MMU). Außerdem muß es natürlich einen Bereich geben, der allen Banken gemeinsam ist. Über diesen Bereich wird die Kommunikation zwischen den Banken abgewickelt.

Eine typische CP/M-Plus-Speicheraufteilung ist in Abb. 5.6 zu sehen. Bank 0 (die während des Kaltstarts aktiv ist) enthält den gebankten Teil des Betriebssystems sowie die Directory-Puffer. Bank 1 steht dem Anwender zur Verfügung, das heißt, hierhin werden die Programme geladen und ausgeführt. Bei jedem Warmstart wird außerdem der Befehls-Interpreter (CCP) in Bank 1 geladen und wie ein Programm ausgeführt. Bank 2 enthält eine Anzahl von Daten-Puffern, die einen wesentlich schnelleren Dateien-Zugriff gestatten.

Adresse	Bank 0	Bank 1	Bank 2
FFFF – F000	gemeinsamer Speicherbereich ungebankter Teil des Betriebssystems		
F000 – 0100	gebankter Teil des Betriebssystems Hash-Tabellen und Directory-Buffer	TPA	Daten-Buffer
0100 – 0000		Base Page	

Abb. 5.6: Typische CP/M-Plus-Speicheraufteilung

SPEICHERAUFTEILUNG BEI MP/M-80

Die Multi-Programming-Fähigkeit von MP/M erfordert ein anderes Speicherkonzept, als es für CP/M üblich ist. Jedes der parallel laufenden Programme hat einen eigenen TPA-Bereich. Zur Bereitstellung mehrerer Bereiche wurden zwei Techniken angewendet:

1. Programme können so generiert werden, daß ihr Laufbereich verschiebbar ist. Sie brauchen dann nicht mehr bei Adresse 100H geladen zu werden. Solche (seiten-)verschiebbaren Programme haben den Dateityp .PRL (page relocatable object).
2. Steht als Speicher mehr als 64 kByte zur Verfügung, wird er (hardwaremäßig) so aufgeteilt, daß mehrere Bänke mit weniger als 64 KByte und ein gemeinsamer Bereich bestehen.

Der Speichplatz wird bei der Systemgenerierung fest in einzelne Ladebereiche (Segmente) aufgeteilt. Die Anzahl dieser Segmente bestimmt auch die maximale Anzahl von parallel arbeitenden Programmen. Abb. 5.7 zeigt die Speicher-Belegung eines typischen gebankten MP/M-Systems.

64 K
gemeinsamer Bereich (common)
residenter Systemteil
48 K
Bank 0
gebankter System-teil
Bank 1
Bank 2
reloc. Segment
32 K
28 K
abs. Segment
abs. Segment
abs. Segment
0

Abb. 5.7: Beispiel einer Speicheraufteilung bei MP/M-80

MULTITASKING

Bei MP/M und CCP/M können von einer Konsole aus mehrere Programme gestartet und dann in den Hintergrund geschaltet werden (Multi Programming). MP/M ermöglicht zudem noch das gleichzeitige Arbeiten von mehreren Konsolen (Multi User). Jedes der Programme kann wiederum aus mehreren Prozessen (mehr oder weniger selbständigen Pro-

grammteilen mit speziellen Aufgaben) bestehen, die (quasi-)parallel arbeiten. Aufgabe des Betriebssystems ist es nun, die Prozessorzeit auf die einzelnen Prozesse aufzuteilen. Diese Aufteilung wird auch als Time-Sharing bezeichnet. Der beim MP/M-System dafür zuständige Programmteil heißt Dispatcher. Die einfachste Art für die Zeitaufteilung ist die Round-Robin-Zuteilung (siehe Abb. 5.8). Die Prozeßzeit wird dabei reihum an die einzelnen Prozesse vergeben. Die einzelnen Prozesse (die Teile des Betriebssystems selbst sind auch in Prozesse aufgeteilt) haben jedoch ein recht unterschiedliches Zeitverhalten. Disk-Zugriffe sollten z.B. möglichst ohne Unterbrechung laufen, während die Druckerausgabe häufig lange Wartezeiten aufweist. Es ist deshalb nützlich, Prozesse mit Prioritäten zu versehen. Das Beispiel einer priorisierten Prozeßverwaltung zeigt Abb. 5.9. Innerhalb einer Prioritätsebene kann natürlich wieder die Round-Robin-Zuteilung angewendet werden. Zur Verwaltung eines Prozesses wird bei MP/M ein Process-Descriptor (PD) verwendet. Er beinhaltet Informationen über den Status des Prozesses und ist über Zeiger mit den anderen PDs verkettet. Der Prozeß kann sich z.B. in einem der folgenden Zustände befinden:

- Prozeß ist bereit und wartet auf Zuteilung von CPU-Zeit
- Prozeß wartet auf Fertigmeldung eines Peripheriegerätes
- Prozeß wartet auf Meldung eines anderen Prozesses
- Prozeß setzt für eine bestimmte Zeit aus

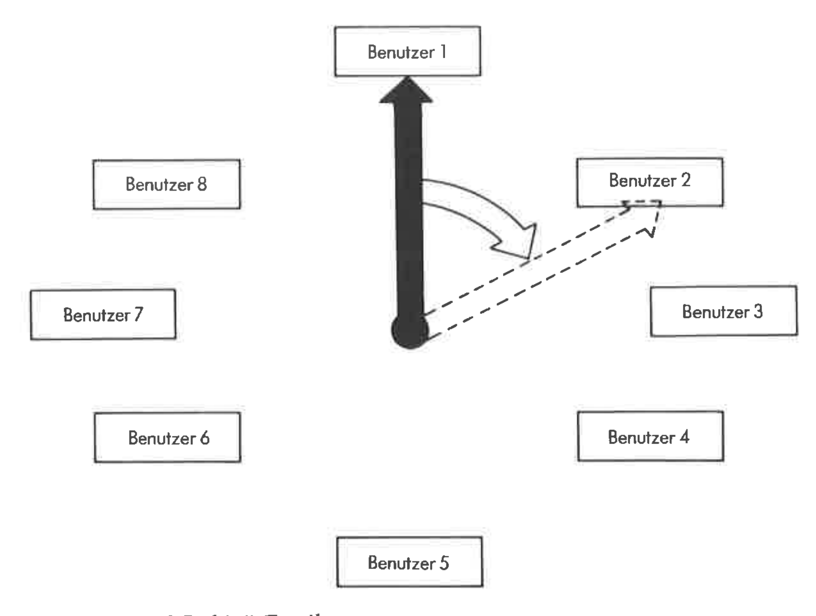


Abb. 5.8: „Round-Robin“-Zuteilung

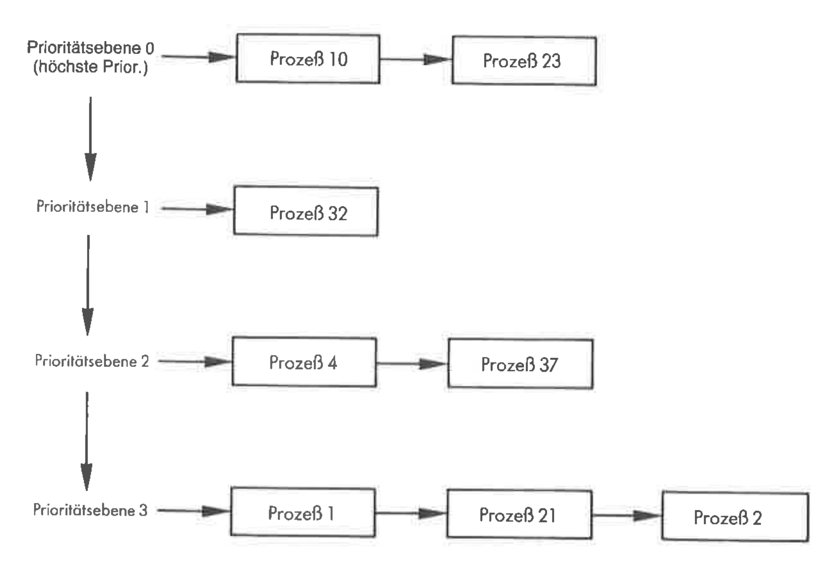


Abb. 5.9: Vier-Ebenen-Prioritätenliste

Der Dispatcher, der die Prozeßzuteilung vornimmt, wird bei MP/M und CCP/M periodisch durch einen Hardware-Interrupt (z.B. alle 16 msec.) sowie bei jedem BDOS-Call aufgerufen. Für bestimmte Teile des Betriebssystems bestehen Sperrlisten, damit sie nicht von mehreren Prozessen gleichzeitig benutzt werden.

Die „Queue" (Warteschlange) ist eine Warteliste, die als spezielle Datei organisiert ist und die geöffnet, gelesen und mit sequentiellen Informationen gefüllt werden kann. Eine „Queue" kann z.B. für die vorübergehende Aufnahme von Informationen genutzt werden, die später von einem anderen Programm zur Diskette übertragen werden.

„Queues" können auch für die sequentiell gesteuerte Ausgabe einzelner Dateien auf dem Drucker verwendet werden. In diesem Fall steht die angesprochene Systemeinheit (Drucker) bis zum Abschluß des Prozesses ausschließlich diesem zur Verfügung (exclusive use of resources). Wenn zum Beispiel ein Prozeß eine „Queue" aufbaut, die nur dann Informationen enthält, wenn der Drucker nicht beschäftigt ist und ein anderer Prozeß vor dem Zugriff auf den Drucker zunächst aus der „Queue" Informationen entnehmen muß, wird durch dieses Verfahren der exklusive Zugriff auf ein gemeinsam benutzbares Betriebsmittel (shared resource) realisiert. Der Beginn des zweiten Prozesses wird dann so lange verschoben, bis er eine Freimeldung des Druckers erhält.

Zur Synchronisierung der einzelnen Prozesse wird noch eine andere Methode unterstützt. Hierbei werden von den einzelnen Prozessen die

von anderen Prozessen gesetzten Zustandsanzeiger (flags) überprüft. Diese Zustandsanzeiger ermöglichen die Synchronisation und Unterbrechung von Prozessen, die von zusätzlichen Hardware-Unterbrechungen und überlagerten Software-Unterbrechungsroutinen vollständig unabhängig sind.

INSTALLATION VON CP/M-80

CP/M ist immer auf eine spezifische Ein/Ausgabe- und Speicherkonfiguration zugeschnitten. Digital Research selbst liefert die Original-CP/M-Versionen für das Entwicklungssystem MDS-800 von Intel.

Einige Hardwarehersteller und Softwarehäuser liefern CP/M-Versionen auch für andere Hardware-Systeme. In den meisten Fällen wird jedoch mit dem Hardware-System eine CP/M-Version zur Verfügung gestellt, die ohne Abänderung sofort einsetzbar ist.

Soll ein verfügbares CP/M-System einer neuen Hardwarekonfiguration angepaßt werden, so müssen Änderungen im BIOS-Modul des CP/M vorgenommen werden. Diese Arbeiten sind zwar nicht extrem schwierig, jedoch geräte- und installationsabhängig. Die spezifischen Instruktionen hierzu werden deshalb hier nicht näher erläutert. Detaillierte Ausführungen zu diesem Thema können im „CP/M Alteration Guide“ von Digital Research nachgelesen werden.

Soll „von Null“ auf eine neue CP/M-Version entwickelt werden, so ist dieses komplexe Problem in kurzer Zeit kaum lösbar. Sind allerdings die Grundelemente für die Entwicklung und Ausführung von Programmen verfügbar, so können eigene Routinen leicht eingebunden (GETSYS und PUTSYS) und die geänderte Version zur System-Diskette zurückgeschrieben oder als neues System auf einer anderen Diskette gespeichert werden.

Liegt ein vollständiges und lauffähiges CP/M vor, dann kann man sehr leicht Assemblerprogramme für spezielle Funktionen erstellen. Auch können die Programme SYSGEN.COM und MOVCPM.COM für die Systemänderung herangezogen werden, so daß keine eigenständigen GETSYS- und PUTSYS-Programme erstellt werden müssen. Allerdings ist es möglich, daß die verfügbare SYSGEN-Version nicht mit den vorhandenen Disketten- oder Plattentypen läuft. In diesem Fall ist eine vorherige Änderung des CP/M unumgänglich.

Digital Research stellt in den Dokumentationsunterlagen (CP/M Alteration Guide oder MP/M Alteration Guide) einige Minimalversionen von GETSYS- und PUTSYS-Programmen bereit.

Zunächst muß ein GETSYS-Programm geschrieben werden, das die ersten beiden Spuren (diese werden mit dem DIR-Befehl nicht angezeigt, da sie Teil des Systems sind) der mitgelieferten Systemdiskette liest. Nun

kann das BIOS-Modul aufgesucht und geändert werden (patching). Das geänderte System kann dann mit einem noch zu schreibenden PUTSYS-Programm gesichert (auf die Diskette geschrieben) werden.

Schließlich kann auch eine eigene Version einer GETSYS-Routine, ein „bootstrap"-Programm, geschrieben und auf Spur 0, Sektor 1 mit dem PUTSYS-Programm abgelegt werden. Nach erfolgtem Test steht ein vollständiges System zur Verfügung, das auch bei einem Kaltstart selbständig startet.

Wenn ein CP/M-System zur Anpassung an eine Hardwarekonfiguration benutzt wird und die neue Konfiguration mit verträglichen Plattenlaufwerken arbeitet, so kann die Systemänderung vergleichsweise schnell mit den Programmen MOVCPM und SYSGEN durchgeführt werden. Diese Systemgenerierung wird in den Handbüchern von Digital Research auch „second level system regeneration" genannt.

Ein Beispiel hierfür ist die Veränderung der Speicheraufteilung mit MOVCPM (anstelle von GETSYS) und die anschließende Implementierung des Systems auf der Systemdiskette mit SYSGEN (anstelle von PUTSYS). Die geänderte Version ist dann dauerhaft auf der Systemdiskette verfügbar.

MOVCPM ist der transiente Befehl zur Datei MOVCPM.COM. Mit dem Befehl müssen Argumente eingegeben werden:

MOVCPM bb *

Hierbei steht bb für die Größe des Arbeitsspeichers in KByte. Mit dem Stern (*) wird dem MOVCPM mitgeteilt, daß der angegebene Speicherbereich im Arbeitsspeicher (TPA) resident zu halten ist. Anstelle der Größe des gewünschten Arbeitsspeichers (bb) kann auch ein zweiter Stern (*) angegeben werden. In diesem Fall wird von MOVCPM der größte Speicherbereich errechnet, über den das System verfügen kann. Im folgenden Beispiel

```
A>MOVCPM 32 *↵
CONSTRUCTION 32K CP/M VERS 2.2
READY FOR "SYSGEN" OR
"SAVE 34 CPM32.COM"
A>
```

wird ein 32 K großes System im TPA generiert (Constructing 32K CP/M VERS 2.2).

Außerdem wird angezeigt, daß das im TPA noch verfügbare System für weitere Verarbeitungen mit einem SYSGEN- oder SAVE-Befehl bereit

steht. Soll dieses System noch für weitere Änderungen verwendet werden, so empfiehlt sich die Sicherung des Systems mit dem SAVE-Befehl (z.B. als CPM32.COM).

Sobald dieses Programm einmal gesichert wurde, kann es jederzeit in den Arbeitsspeicher geladen und das BIOS z.B. mit dem CP/M-Debugger DDT.COM geändert werden.

Wenn umfangreiche Änderungen vorgenommen werden sollen, so ist eine Neuerstellung des BIOS (z.B. unter dem Namen CBIOS) effektiver. Auf diese Weise kann dann auch ein eigenes Bootstrap-Programm (z.B. unter dem Namen BOOT) generiert werden. Für die Erstellung der notwendigen Quellprogramme (BIOS.ASM und BOOT.ASM) kann dann der CP/M-Editor ED.COM und für die Assemblierung der CP/M-Assembler ASM.COM benutzt werden.

Die so assemblierten Programme CBIOS.HEX und BOOT.HEX können mit dem LOAD-Befehl in „.COM"-Programme transformiert und danach vor dem Einbinden in das System einzeln getestet werden. Wenn das neue System CPMbb.COM (bb ist die Speichergröße im MOVCPM) bereits generiert ist, kann es mit dem DDT in den Arbeitsspeicher geladen und dort mit den Modulen (BIOS.COM und BOOT.COM) verbunden, getestet und ausgetauscht werden. Mit der SYSGEN-Routine wird das so erhaltene System auf die ersten beiden Spuren der neuen Systemdiskette geschrieben.

Folgendes Beispiel beschreibt diesen Vorgang:

```
A>SYSGEN↵
SYSGEN VERSION 2.0
SOURCE DRIVE NAME (OR RETURN TO SKIP)↵
                (Antwort mit RETURN. Hierdurch wird das Einle-
                sen des Systems verhindert. Das geänderte
                System befindet sich ja schon im Arbeitsspeicher.)
DESTINATION DRIVE NAME (OR RETURN TO REBOOT)B
                (Hierdurch wird eingegeben, daß das System auf
                die neue Diskette in Laufwerk B geschrieben wer-
                den soll.)
DESTINATION ON DRIVE B, THEN TYPE RETURN↵
FUNCTION COMPLETE
DESTINATION DRIVE NAME (OR RETURN TO REBOOT)↵
A>PIP B: = A:*.*[V]↵
.....
                (Meldungen einzelner Kopiervorgänge)
.....
A>
```

Wenn bereits eine Kopie des CP/M-Systems als „.COM“-Datei auf der Diskette abgelegt ist, kann der oben beschriebene Vorgang wie folgt abgekürzt werden:

```
A>SYSGEN CPM.COM↲
SYSGEN VERSION 2.2
DESTINATION DRIVE NAME (OR RETURN TO REBOOT):
.
.
.
```

Für Änderungen innerhalb des BIOS müssen die einzelnen Anweisungen im Handbuch „CP/M Version 2.2 Alternative Guide“ beachtet und hiernach die benötigten GETSYS-, PUTSYS- und BOOT-Programme generiert werden.

REKONFIGURATION (Anpassung der Speichergröße mit MOVCPM)

Oftmals wird bei einem Hardware-System die Größe des Arbeitsspeichers erhöht oder erniedrigt (herausnehmen von Speicherkarten). Entsprechend muß dann auch das CP/M geändert werden. Dazu dient das Programm MOVCPM, das das System (allerdings normalerweise mit dem MDS-800-BIOS) für den entsprechenden Adreßraum herstellt. Das MOVCPM-Programm kann wie ein Befehl mit zusätzlichen möglichen Argumenten gestartet werden:

```
         { *  } {   }
MOVCPM   {    } { * }
         { bb } {   }
```

Das wahlweise mit bb angegebene Argument teilt dem MOVCPM-Programm mit, welche Speichergröße im neuen CP/M verwaltet werden soll. Wird dieses Argument weggelassen oder durch einen Stern * bezeichnet, so wird der gesamte verfügbare RAM-Bereich (RAM = Random Access Memory) des Computersystems konfiguriert. In den meisten Fällen wird gewünscht, daß vom CP/M-System der gesamte Speicherbereich verwaltet wird.

Der zweite wahlweise angebbare Stern (*) teilt dem MOVCPM-Programm mit, daß das neu konfigurierte System im Arbeitsspeicher (TPA) für einen folgenden SYSGEN- oder SAVE-Befehl verbleiben soll. Häufig wird das neue System mit SAVE gesichert oder mit SYSGEN auf eine Diskette geschrieben.

Wird der zweite * nicht angegeben, so wird das neue System ohne vorherige Sicherung geladen.

Hier einige Beispiele:

A>**MOVCPM**↵ — Mit diesem Befehl wird das CP/M so geändert (und ohne Abspeicherung auf der Diskette gestartet), daß der gesamte RAM-Bereich des Rechners ausgenutzt wird (Anfang des TPA ab 100H).

A>**MOVCPM 32**↵ — Mit diesem Befehl wird ein CP/M-System für die Verwaltung von 32K Arbeitsspeicher erstellt und ohne Sicherung auf Diskette gestartet.

A>**MOVCPM 32** *↵ — Mit diesem Befehl wird ein CP/M-System für die Verwaltung von 32K Arbeitsspeicher erstellt und im TPA-Bereich für die Anwendung von SYSGEN oder SAVE belassen.

A>**MOVCPM** * *↵ — Dieser Befehl erstellt ein CP/M-System, das den gesamten Arbeitsspeicherbereich des Rechners (beginnend von 100H) ausnutzt und dieses System für die nachfolgende Anwendung des SYSGEN- oder SAVE-Befehls im TPA-Bereich beläßt.

Die beiden letzten Formen benutzen den zweiten Stern, um den Verbleib des Systems im TPA-Bereich zu veranlassen. Mit dem SAVE-Befehl kann dieses dann als Datei zur Diskette oder mit dem SYSGEN-Befehl als neues CP/M-System auf die ersten beiden Spuren einer Diskette geschrieben werden.

Immer wenn MOVCPM * *↵ oder MOVCPM bb *↵ ausgeführt wird, meldet das Programm „READY FOR SYSGEN" oder „SAVE bb CPMbb.COM", um so auf die möglichen Folgeoperationen aufmerksam zu machen (bb entspricht der Anzahl KBytes).

AUTO-START BEI CP/M-80

Die Befehlszeile, die nach jedem Rücksprung in das System aufgerufen wird (angedeutet mit der Systembereitschaftsmeldung), kann mit einem geeigneten „Menü"-Programm versehen werden. Dieses „Menü"-Programm erzeugt die Auflistung einer Programmauswahl, von wo aus durch eine einzelne Auswahlentscheidung das gewünschte Programm aufgerufen wird und wo nach Ausführung wieder das Auswahlmenü automatisch

auf dem Bildschirm erscheint. Bei einem solchen CP/M-System können dann allerdings nur die im Auswahlmenü aufgelisteten Programme bzw. Befehle gestartet werden, weil ein Rücksprung in das System (^C) immer wieder das Menü-Programm startet.

Zur Generierung einer eigenen Befehlszeile für Programmauswahl und automatischen Programmstart muß zunächst das System mit MOVCPM oder SYSGEN in den Speicherbereich und danach als Datei mit dem SAVE-Befehl abgespeichert werden. Danach kann mit DDT die gewünschte Änderung durchgeführt werden. Die Startadresse für die Änderung ist 0980H. Mit dem D-Befehl im DDT kann der Inhalt dann ab dieser Speicherstelle angezeigt werden:

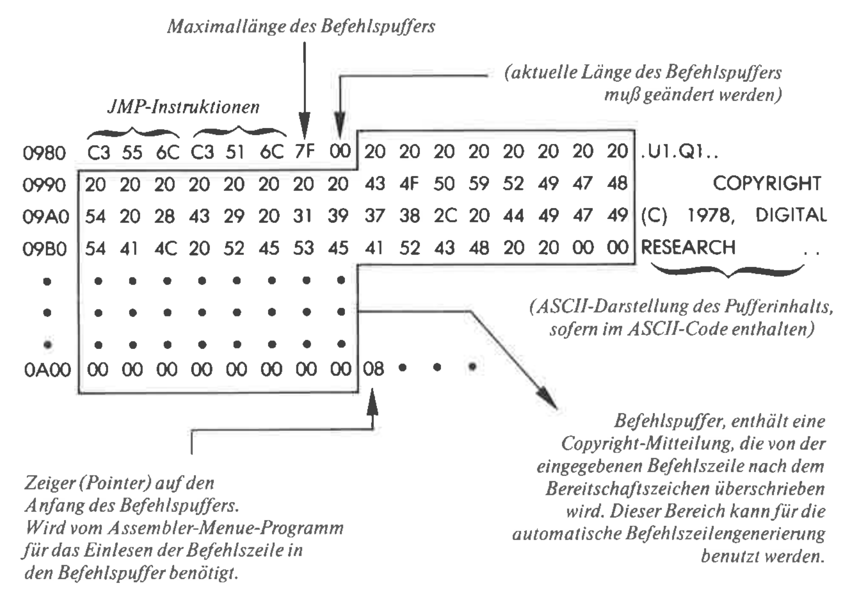


Abb. 5.10: Anzeige des Speicherinhalts

BEISPIEL EINER CP/M-ÄNDERUNG: MENÜ-SYSTEM

Dieses Beispiel soll zeigen, wie die einzelnen CP/M-Möglichkeiten effektiv genutzt werden können. Von einem wirkungsvollen System für anspruchsvolle Aufgaben im technischen und kommerziellen Bereich wird im allgemeinen erwartet, daß es ohne zusätzliche Systemkenntnisse eingesetzt werden kann. Hierzu gehören die „Turn-Key"-Systeme, die nach einem Kaltstart oder nach Drücken der Reset-Taste automatisch das gewünschte Programm laden und ausführen. Eine andere Form

anspruchsvoller Anwendungen wird von Systemen abgedeckt, die automatisch in einem „Turn-Key"-System ein Menü-Programm starten, von dem aus leicht das gewünschte Programm ausgewählt werden kann.

Beide Formen können durch die Modifikation spezieller Bereiche im CP/M-System erreicht werden. Diese Bereiche sind dort zu finden, wo normalerweise nach der Bereitschaftsmeldung ein Befehl eingegeben wird. Der Änderungsbereich beginnt mit der Anfangsadresse des CCP (Console Command Processor).

Mit dem DDT (oder jedem anderen Debugger, wie z.B. SID) können die einzelnen Speicherstellen leicht geändert werden. Dabei muß an dieser Stelle eine Befehlszeile eingefügt und ein Zustandsbit (flag) gesetzt werden, um damit dem CP/M zu signalisieren, daß anstelle der Angabe einer Bereitschaftsmeldung die Befehlszeile zu bearbeiten ist. Wenn später ein Benutzer einen Kaltstart oder Warmstart durchführt, wird die eigentliche Befehlszeile automatisch ausgeführt.

Die angezeigten Werte der einzelnen Befehle sind Hexadezimalwerte. Jedem ASCII-Zeichen ist ein bestimmter Hexadezimalwert zugeordnet. So hat das C in COPYRIGHT den Wert 43H (H steht für hexadezimal), das O in COPYRIGHT den Wert 4FH und ein Leerzeichen den Wert 20H (die ersten Zeichen im Befehlspuffer von Abb. 5.10). Die Werte können aus einer ASCII-Tabelle leicht ermittelt werden.

Zu beachten ist, daß für die aktuelle Befehlspufferlänge (Position in Adreßzeile 0980H hinter dem Wert für die maximale Befehlspufferlänge) der Wert 00 gesetzt ist. Dieser Wert teilt dem CP/M nach einem Warm- oder Kaltstart mit, daß im Befehlspuffer kein Befehl steht. Das System meldet sich in diesem Fall mit dem Bereitschaftszeichen A> und wartet auf einen Befehl vom Benutzer, der in den Befehlspuffer durch Überschreiben des vorherigen Inhalts eingelesen wird (bei einem Kaltstart belegt die Copyright-Mitteilung noch den Befehlspuffer).

Für die Realisierung eines „Turn-Key"-Systems muß von diesem normalen Systemstart abgewichen werden. Wird die aktuelle Länge des Befehlspuffers in einen von Null verschiedenen Wert geändert, so nimmt das System nach einem Kalt- oder Warmstart an, daß bereits ein Befehl im Befehlspuffer steht, und führt diesen aus. Nach jeder vollständigen Ausführung oder nach einer Befehlsunterbrechung (z.B. durch Kalt- oder Warmstart) wird vom System immer wieder angenommen, daß im Befehlspuffer bereits ein Befehl steht, so daß niemals mehr das Bereitschaftszeichen A> ausgegeben wird.

Die Befehlszeile kann über die Copyright-Mitteilung geschrieben werden (die Copyright-Mitteilung kann jedoch auch an das Ende des Befehlspuffers geschrieben werden). Erlaubt ist ein Beschreiben des Befehlspuffers bis einschließlich Position 0A07H. Darüber hinaus darf keine Informa-

tion geändert werden. Position 0A08H enthält den Zeiger für den Beginn des Befehlspuffers, der z.B. noch in dem Menü-Programm (im Arbeitsspeicher) benötigt wird.

Eine Befehlszeile läßt sich mit dem S-Befehl im DDT einfügen. Hierbei müssen die Hexadezimalwerte für die gewünschten ASCII-Zeichen der Befehlszeile eingegeben werden. Diese Befehlszeile sollte mit dem Wert 00 abgeschlossen werden. Danach muß dann noch der Wert für die aktuelle Länge des Befehlspuffers eingegeben werden. Er errechnet sich aus der Anzahl aller in der Befehlszeile bis zum abschließenden 00-Wert auftretenden Zeichen, einschließlich aller Leerzeichen. Dieser Wert teilt dem CP/M die Länge der Befehlszeile mit.

Eine einfache Möglichkeit des Aufbaus eines Menü-Systems ist die Anwendung eines BASIC-Interpreters. Die meisten BASIC-Versionen verfügen heute bereits über CHAIN-Anweisungen, mit denen unabhängige BASIC-Programme verkettet aufgerufen und gestartet werden können und von denen auch das Menü-Programm wieder gestartet werden kann.

Das Menü-Programm kann dann in BASIC programmiert werden. Hierdurch erübrigt sich außerdem noch die Berücksichtigung des Befehlspuffer-Pointers (zeigt auf Position 0A08H), da dieser automatisch vom BASIC-Interpreter verwaltet wird.

Als BASIC-Versionen bieten sich CBASIC2 (kommerziell einsetzbares BASIC-Softwaresystem) und das MICROSOFT-BASIC (MBASIC. COM) an. Beide erlauben von der CP/M-Befehlszeile aus das Einladen und Starten von BASIC-Programmen.

Das MICROSOFT-BASIC wird als MBASIC.COM geliefert und erlaubt z.B. sowohl den Aufruf des BASIC-Interpreters (durch MBASIC↵) als auch das zusätzliche Laden und Starten eines Anwenderprogramms, das durch ein Argument in der Befehlszeile mit angegeben wird. In dem folgenden Befehl

```
A>MBASIC PROG↵
```

ist MBASIC der BASIC-Interpreter, der das BASIC-Programm PROG. BAS lädt und ausführt. Vom MICROSOFT-BASIC wird der Dateityp „BAS“ erwartet.

Wird nun ein BASIC-Programm MENUE.BAS geschrieben, so müßte die Befehlszeile den Dateinamen dieses Menü-Programms aufweisen:

```
MBASIC MENUE
```

Diese Befehlszeile kann in den Befehlspuffer eingefügt werden. Die aktuelle Länge des Befehlspuffers beträgt somit 12 Positionen und muß als Hexadezimalwert 0C an die Position 0987H vor die Befehlszeile MBASIC MENUE in den Befehlspuffer geschrieben werden (vgl. Abb. 5.11).

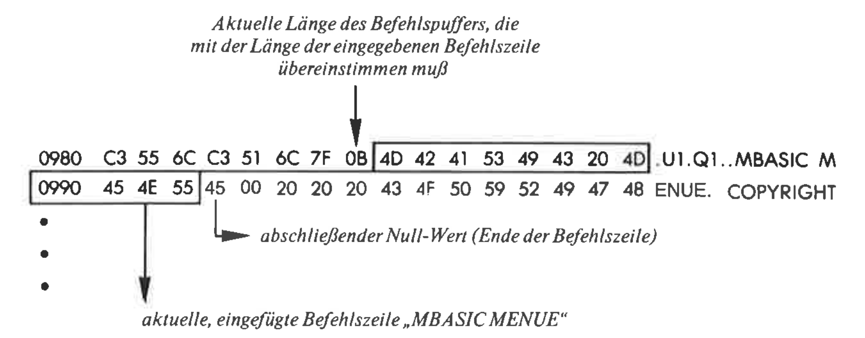


Abb. 5.11: Speicherausschnitt mit eingefügter Befehlszeile

Nach Abschluß der Einfügungen und Änderungen (patching) muß der GO-Befehl im DDT eingegeben werden, der die DDT-Routine abbricht. Der anschließend auszuführende SAVE-Befehl speichert den Inhalt des TPA-Bereichs als neue Systemversion ab.

```
A>SAVE 34 AUTOCPM.COM↵
```

(Mit „AUTOCPM.COM" läßt sich diese Version von anderen CP/M-Versionen leicht unterscheiden.)

```
A>SYSGEN↵
SOURCE DRIVE NAME (OR RETURN TO SKIP)↵
            (Antwort mit RETURN. Das geänderte System
            befindet sich bereits im Arbeitsspeicher.)
DESTINATION DRIVE NAME (OR RETURN TO REBOOT)B
            (die neue Systemdiskette soll auf Laufwerk B
            erstellt werden)
DESTINATION IN DRIVE B, THEN TYPE RETURN↵
FUNCTION COMPLETE
DESTINATION DRIVE NAME (OR RETURN TO REBOOT)↵
            (Eingabe von RETURN zum Verlassen der
            SYSGEN-Routine)
A>
```

Nun müssen nur noch alle benötigten Dateien (CP/M-Befehlsdateien, MBASIC.COM und MENUE.BAS) auf die neue Systemdiskette kopiert werden.

Bei einem Kaltstart führt CP/M dann automatisch MBASIC mit dem Programm MENUE.BAS aus, ohne vorher oder nachher die Bereitschaftsmeldung auszugeben. Das Programm MENUE.BAS muß die Zugriffsmöglichkeit auf andere BASIC-Programme ermöglichen (über CHAIN oder SWAP).

Soll das Menü-Programm in Assemblersprache geschrieben werden, so muß es in der Lage sein, eine weitere Befehlszeile im Befehlspuffer für die Verzweigung in ein anderes Programm zu generieren. Dieses Programm muß auf den Zeiger des Pufferanfangs in Position 0A08H zurückgreifen und dem System mitteilen, daß es wieder an den Pufferanfang zurückspringen und die neue Befehlszeile ausführen soll.

INSTALLATION VON CP/M-PLUS

Bei CP/M-Plus steht nicht mehr das komplette System, sondern nur noch der Systemlader auf den Systemspuren (wenn nicht vom Standard-CP/M aus gestartet werden soll). Das eigentliche Betriebssystem steht als Datei CPM3.SYS auf der Systemplatte. Dieser File wird von einem Systemgenerierungs-Programm (GENCPM.COM) erstellt. Das Generierungsprogramm stellt alle benötigten Teile des Systems zusammen und setzt, im Dialog mit dem Programmierer, die erforderlichen Parameter. Die möglichen Konfigurationen bei CP/M-Plus können sehr unterschiedlich sein, so daß wir hier nicht weiter auf die Installation eingehen werden. Ein wichtiger Unterschied zum Standard-CP/M ist, daß nur noch der CCP beim Warmstart nachgeladen wird, und zwar auf Adresse 100H. (Deshalb gibt es hier auch keinen residenten SAVE-Befehl.)

INSTALLATION VON CP/M-86

Die von Digital Research vertriebene Standard-Version des CP/M-86 ist für den Intel SBC 86/12 Single-Board-Computer vorbereitet und steht auf einer 8-Zoll-Single-Density-Diskette. Eine Systemanpassung vom „Null-Stand" auf andere Rechner ist ohne weitreichende Sachkenntnisse nicht möglich, allerdings kann die Möglichkeit, die Vorbereitungen auf einem CP/M-80 System zu machen, hilfreich sein.

Anders als bei CP/M-80 steht das Betriebssystem nicht auf den Systemspuren (dort steht nur ein Ladeprogramm), sondern als Datei CPM.SYS im Datenteil der Platte. Nach dem Kaltstart, bei dem es vom Lader in den Speicher gebracht wird, bleibt es dort resident, das heißt, es werden keine Teile beim Warmstart nachgeladen.

Die CPM.SYS-Datei besteht aus dem BDOS und dem Rechner-angepaßten BIOS. Nach Änderung des BIOS wird auf folgende Weise eine neue CPM.SYS-Datei gebildet:

```
A>PIP CPMX.H86 = CPM.H86,BIOS.H86↵
A>GENCMD CPMX 8080 CODE[A40]↵
A>REN CPM.SYS = CPMX.CMD↵
A>
```

INSTALLATION VON MP/M-80 und MP/M-86

Das MP/M-System ist zu umfangreich, um auf den Systemspuren untergebracht zu werden. Hier befindet sich das System als Datei unter dem Namen MPM.SYS im Datenteil der Systemplatte. Mit dem Programm MPMLDR wird es dann in den Speicher gebracht und gestartet. Das Ladeprogramm MPMLDR kann entweder von einem zuvor im Speicher befindlichen CP/M gestartet werden oder aber auf den Systemspuren stehen. Es enthält ein eigenes, stark abgemagertes CP/M einschließlich BIOS. Dieses LDRBIOS muß natürlich für den Rechner angepaßt werden. Zur Generierung der MPM.SYS-Datei dient ein spezielles Generierungsprogramm: GENSYS. Es verbindet alle zum Betriebssystem nötigen Teile (die als Dateien auf der Platte sein müssen) einschließlich des Hardware-abhängigen XIOS.
Im MP/M werden keine „built-in"-Befehle verwendet. Alle Befehle sind entweder transiente „.COM"-, „.PRL"- bzw. „CMD"-Programme oder die während der Systemgenerierung aus den „RSP"-Programmen erzeugten residenten Systemprozesse.

Das folgende Beispiel zeigt eine System-Generierung für ein gebanktes MP/M-80-System mit 2 Arbeitsplätzen:

```
0A>GENSYS↵

MP/M II V2.1 System Generation
Copyright © 1981, Digital Research

Default entries are shown in (parens).
Default base is Hex, precede entry with # for decimal

Use SYSTEM.DAT for defaults (Y) ?
Top page of operating system (FF) ?
Number of TMPs (system consoles) (#2) ?
Number of Printers (#1) ?
Breakpoint RST (06) ?
```

```
Enable Compatibility Attributes (N) ?
Add system call user stacks (Y) ?
Z80 CPU (Y) ?
Number of ticks/second (#60) ?
System Drive (A:) ?
Temporary file drive (M:) ?
Maximum locked records/process (#16) ?
Total locked records/system (#32) ?
Maximum open files/process (#16) ?
Total open files/system (#32) ?
Bank switched memory (Y) ?
Number of user memory segments (#3) ?
Common memory base page (C0) ?
Dayfile logging at console (Y) ?

    SYSTEM    DAT  FF00H  0100H
    TMPD      DAT  FE00H  0100H
    USERSYS   STK  FD00H  0100H
    XIOSJMP   TBL  FC00H  0100H

Accept new system data page entries (Y) ?

    RESBDOS   SPR  F000H  0C00H
    XDOS      SPR  CE00H  2200H

Select Resident and Banked System Processes:
    MPMSTAT   RSP  (N)   ?

    BNKXIOS   SPR  C200H  0C00H
    BNKBDOS   SPR  9F00H  2300H
    BNKXDOS   SPR  9D00H  0200H
    TMP       SPR  9900H  0400H

    LCKLSTS   DAT  9600H  0300H
    CONSOLE   DAT  9400H  0200H

Enter memory segment table:

    Base,size,attrib,bank (00,C0,00,01) ?
    Base,size,attrib,bank (00,C0,00,02) ?
    Base,size,attrib,bank (00,C0,00,03) ?

    MP/M II   Sys  9400H   6C00H  Bank 00
    Memseg    Usr  0000H   C000H  Bank 01
    Memseg    Usr  0000H   C000H  Bank 02
    Memseg    Usr  0000H   C000H  Bank 03
```

```
Accept new memory segment table entries (Y) ?

** GENSYS DONE **

0A>
```

GENSYS legt außer der System-Datei MPM.SYS noch eine Datei mit den Angaben für die System-Generierung an: SYSTEM.DAT. Diese Datei kann bei zukünftigen Generierungen als Ausgangsbasis dienen.

Top page ist die höchste 256-Byte-Seite des verfügbaren Speichers (bei 64 KByte also FF Hex).

Die *Anzahl der TMPs* (Terminal-Prozesse) gibt die Anzahl der Arbeitsplätze am System an.

Verträglichkeits-Attribute werden zum definierten Ausschalten des Dateischutzes benötigt.

Die *Breakpoint RST*-Nummer wird bei der Benutzung von Debuggern (z.B. DDT.PRL) in seitenweise verschiebbaren Speicherbereichen benötigt.

System call user stacks sollten bei Benutzung von CP/M-Programmen zur Sicherstellung der Verträglichkeit vereinbart werden.

Bei Verwendung von *Z80*-Programmen, die die speziellen Z80-Register benutzen, müssen diese vom Dispatcher mit verwaltet werden.

Die Angabe von *ticks/second* stellt die Frequenz des System-Taktes (ein zyklischer Interrupt) dar und wird z.B. bei programmierten Zeitverzögerungen benötigt.

Das Systemlaufwerk ist das Laufwerk, von dem ein Programm geladen wird, das auf dem Arbeitslaufwerk nicht gefunden wurde. Das *Temporary file drive* ist ein Laufwerk, auf dem nur kurzzeitig existierende Dateien gespeichert werden (z.B. von SUBMIT). Häufig wird hier eine sogenannte RAM-Floppy eingesetzt: die Nachbildung eines Laufwerks mit Halbleiterspeicher.

Die Angaben über die Anzahl von *locked records* und *open files* dienen zur Bereitstellung von Speicherplatz für den Schutz von Dateien bei Zugriffen von mehreren Programmen gleichzeitig.

Bank switched memory gibt das Speichermodell an, das für Aufteilung des Speicherraums wichtig ist. Die Anzahl der Speichersegmente entspricht der maximalen Anzahl von parallel laufenden Programmen. Die *Common memory base page* ist die Speicherseite, ab der der Speicherbereich für alle Banks gemeinsam ist.

Das *Dayfile logging* bewirkt eine Zeitangabe bei jedem Programmaufruf.

Das GENSYS-Programm sucht auf der Arbeitsplatte nach allen Dateien vom Typ „.RSP“ und fragt jeweils, ob sie mit in das System eingebunden werden soll.

Die Adressen der Speicher-Segmente werden in 256-Byte-Seiten angegeben. Das erste Speichersegment ist reserviert für das System. Beachtet werden muß, daß „.COM“-Programme nur in Speichersegmenten lauffähig sind, die bei Seite 00 beginnen.

ANPASSUNG VON CCP/M

CCP/M wird z.Z. von Digital Research in zwei Versionen geliefert: zum einen fertig angepaßt für den IBM-PC und zum anderen als „Generic Version“. Letztere kann als Basis für eine Anpassung an eine spezielle Hardware verwendet werden.

ZUSAMMENFASSUNG

In diesem Kapitel wurden die Struktur und die internen Abläufe in den Betriebssystemen der CP/M-Familie erklärt. Die Grundlagen sind nicht sehr komplex, doch wird der System-Programmierer häufig nicht ohne zusätzliche Informationen aus den Manualen von Digital Research auskommen. Außerdem ist eine ausreichende Kenntnis der Rechnerhardware unabdingbar.

Kapitel 6

Kurzbeschreibungen der Befehle und Programme

EINFÜHRUNG

Dieses Kapitel enthält eine alphabetisch geordnete Zusammenstellung der zum Lieferumfang der CP/M-Familie gehörenden Befehle und Programme. Die Beschreibungen sind nach einem gleichbleibenden Schema aufgebaut. Im Kopf jeder Beschreibung sind die Systeme aus der CP/M-Familie markiert, für die sie Gültigkeit hat. Berücksichtigt sind dabei folgende Systeme:

- CP/M-80 die Version 2.2 des CP/M für die 80er Prozessoren
- CP/M-Plus Version 3 von CP/M für die 80er Prozessoren
- MP/M-80 Version 2 für die 80er Prozessoren
- CP/M-86 CP/M-Version für die 86er Prozessoren
- CCP/M-86 Concurrent CP/M-86 für die 86er Prozessoren
- MP/M-86 MP/M für die 86er Prozessoren

Im folgenden Beispiel wird durch den ausgefüllten Kreis bei der Version CP/M-Plus angezeigt, daß der zugehörige Befehl nur in diesem System vorhanden ist:

o CP/M-80	• CP/M-86
o CP/M-Plus	o CCP/M
o MP/M-80	o MP/M-86

Die Beschreibung selbst ist gegliedert in Format, Argumente, Beschreibung, Anwendung und Beispiele. Argumente, die optional sind (also auch wegfallen können), sind *kursiv* gedruckt. Die geschweiften Klammern { } zeigen eine Auswahlmöglichkeit an, wobei ein Argument immer dann zwingend notwendig ist, wenn es nicht *kursiv* angegeben ist. In einigen Fällen ist ein Teil eines Arguments wahlweise, während ein anderes zwingend notwendig ist (z.B. d:filename, wobei d: fakultativ und filename zwingend ist). In jedem Fall sollte die Beschreibung der Argumente mit herangezogen werden.

Weiterhin sind einige generelle Konventionen unterstellt. So ist zum Beispiel vereinbart, daß Dateiargumente, ob sie benötigt werden oder nicht,

in jedem Fall eine Laufwerksbezeichnung enthalten können (z.B. B:DATEI). Hiermit wird angegeben, daß auch ein anderes als das selektierte Laufwerk angesprochen werden kann. Wegen dieser generellen Vereinbarung sind die Laufwerkssymbole in den Befehlsbeschreibungen weggelassen worden. Enthält eine Beschreibung ein Laufwerkskennzeichen, so soll hiermit eine besondere Abmachung angesprochen werden. In diesem Fall sind in der Argumentbeschreibung weitere Erläuterungen angegeben.

Alle Zeichen, die unterstrichen sind, sind für die Eingabe des Befehls erforderlich. Das Symbol ↵ steht für das Drücken der Taste RETURN oder CR (carriage return). Das Symbol ^ mit einem nachfolgenden Buchstaben steht für ein Steuerzeichen (CTRL-Zeichen), das durch gleichzeitiges Drücken der CTRL-Taste mit einer Buchstaben-Taste (z.B. ^C) erzeugt wird.

- o CP/M-80
- o CP/M-Plus
- • MP/M-80
- o CP/M-86
- • CCP/M
- • MP/M-86



ABORT

Abbruch eines laufenden Programms

Format:

1. ABORT programmname
2. ABORT programmname konsolnummer

Argumente:

programmname	Der Name des laufenden Programms, das unterbrochen werden soll.
konsolnummer	Nummer der Konsole, von der das laufende Programm gestartet wurde. Sie braucht nur eingegeben zu werden, wenn das Programm von einer anderen Konsole gestartet wurde.

Beschreibung:

Dieser Befehl unterbricht die Ausführung des spezifizierten Programms. Er sollte sehr vorsichtig eingesetzt werden, da hiermit jedes beliebige Programm eines jeden Benutzers unterbrochen werden kann. Residente System-Prozesse können hiermit nicht abgebrochen werden.

Anwendung:

Wenn das Programm von der eigenen Konsole gestartet und dann in den Hintergrund geschaltet (detached) wurde, braucht nur ABORT und der Programmname eingegeben zu werden; wurde es von einer anderen Konsole gestartet, so muß deren Nummer mit angegeben werden. Kann aus irgendeinem Grund nicht abgebrochen werden (z.B. die Konsolnummer stimmt nicht), so erscheint die Meldung:

```
Abort failed
```

Beispiele:

```
0A>ABORT COMPUTE↵
```

(COMPUTE war ursprünglich von dieser Konsole aus gestartet worden)

```
2A>ABORT TEST 1↵
```

(TEST war von Konsole 1 aus gestartet worden)

- ● CP/M-80
- o CP/M-Plus
- ● MP/M-80
- o CP/M-86
- o CCP/M
- o MP/M-86

ASM

8080-Assembler

Format:

1. ASM dateiname
2. ASM dateiname.*shp*

Argumente:

dateiname	Der Name einer Quelldatei in Assemblersprache. Der Dateityp „.ASM“ wird vorausgesetzt.
.shp	In diesem Fall steht hier nicht der Dateityp, sondern drei Zeichen als optionale Parameter. s, h und p werden dabei in folgender Weise ersetzt:
s = A...P	Angabe des Laufwerks, auf dem sich die Quelldatei (source) befindet.
h = A..P,Z	Angabe des Laufwerks, auf dem die Objektdatei (mit Dateityp „.HEX“) geschrieben werden soll. Z unterdrückt die Generierung der Objektdatei.
p = A..P,X,Z	Angabe des Laufwerks, auf dem das Listing (als „.PRN“-Datei) geschrieben werden soll. X leitet das Listing zur Konsole, Z unterdrückt seine Generierung.

Beschreibung:

Der Assembler (ASM.COM) generiert aus einer Quelldatei mit Assemblerbefehlen für den 8080-Mikroprozessor eine Zieldatei im Maschinencode in hexadezimaler Darstellung. Gleichzeitig kann ein sogenanntes Listing erzeugt werden, das neben dem Quellcode auch den übersetzten Code und ggf. Fehlerkennzeichnungen enthält.

Anwendung:

Der Assembler erfordert als Quelle eine Textdatei, die die syntaktischen Regeln der 8080-Assemblersprache einhält. So eine Datei kann z.B. mit

dem Editor erstellt werden. Sie besitzt den Dateityp „.ASM“. Zum Aufruf des Assemblers kann, wenn nur auf dem selektierten Laufwerk (das ist das Laufwerk, das der Befehls-Interpreter in seiner Meldezeile anzeigt) gearbeitet werden soll, das Format 1 benutzt werden. Soll auf andere Laufwerke zugegriffen oder eine Ausgabe unterdrückt werden, so muß Format 2 gewählt werden. Der generierte Objektcode ist noch nicht direkt ausführbar, sondern wird in einer lesbaren Form im sogenannten Intel-Hex-Format ausgegeben. Zur Umwandlung in eine binäre Form dient das Programm LOAD.

Beim CP/M-86, MP/M-86 und CCP/M gehört der Assembler ASM86 zum Lieferumfang.

Beispiele:

A>**ASM PROG**↵

(ASM assembliert die Datei PROG.ASM und generiert die Dateien PROG.HEX und PROG.PRN. Alle 3 Dateien stehen auf Laufwerk A.)

A>**ASM TUDAS.ABZ**↵

(ASM assembliert die Datei TUDAS von Laufwerk A und erzeugt die Datei TUDAS.HEX auf Laufwerk B. Ein Listing wird nicht generiert.)

o CP/M-80	• CP/M-86	
o CP/M-Plus	• CCP/M	
o MP/M-80	• MP/M-86	



8086-Assembler

Format:

ASM dateiname $parameter

Argumente:

dateiname — Der Name einer Quelldatei in Assemblersprache. Wird der Dateityp weggelassen, wird automatisch „.A86“ angenommen.

parameter — Liste von Parametern, bestehend aus jeweils 2 Buchstaben, wobei der zweite ein Gerät bezeichnet. Die Bedeutung der jeweils ersten Zeichen ist folgende:

- A — Es folgt die Angabe des Quell-Laufwerks (A...P)
- H — Es folgt die Geräte-Bezeichnung zur Ausgabe des Hex-Objekt-Codes.
- P — Es folgt die Geräte-Bezeichnung zur Ausgabe des Assembler-Listings.
- S — Es folgt die Geräte-Bezeichnung zur Ausgabe der Symboltabelle.

Gerätebezeichnungen können neben den Laufwerken A...P auch folgende sein:

- X — Terminal
- Y — Printer
- Z — keine Ausgabe (zero output)

Beschreibung:

Der Assembler (ASM86.CMD) generiert aus einer Quelldatei mit Assemblerbefehlen für den 8086-Mikroprozessor eine Zieldatei mit Maschinencode in hexadezimaler Darstellung. Gleichzeiting kann ein sogenanntes Listing erzeugt werden, das neben dem Quellcode auch den übersetzten Code und ggf. Fehlerkennzeichnungen enthält.

Anwendung:

Der Assembler erfordert als Quelle eine Textdatei, die die syntaktischen Regeln der 8086-Assemblersprache einhält. So eine Datei kann z.B. mit dem Editor erstellt werden. Der generierte Objektcode ist noch nicht direkt ausführbar, sondern wird in einer lesbaren Form im sogenannten Intel-Hex-Format ausgegeben. Zur Umwandlung in eine binäre Form dient das Programm GENCMD.

Zur Systemprogrammierung auf einem CP/M-80-System steht auch eine Cross-Assembler-Version zur Verfügung. Dieser Assembler (ASM86. COM) läuft auf dem 80er-System und erzeugt Codes für das 86er-System.

Beispiele:

```
A>ASM86 PROG↲
```

(ASM86 assembliert die Datei PROG.A86 und generiert die Dateien PROG.H86, PROG.PRN und PROG.SYM. Alle 4 Dateien stehen auf Laufwerk A.)

```
A>ASM86 TUDAS $AA HB PY SX↲
```

(ASM86 assembliert die Datei TUDAS.H86 von Laufwerk A und generiert die Datei TUDAS.H86 auf Laufwerk B. Das Listing wird auf den Drucker ausgegeben, die Symboltabelle auf den Bildschirm.)

- ○ CP/M-80
- ○ CP/M-Plus
- ● MP/M-80
- ○ CP/M-86
- ○ CCP/M
- ● MP/M-86



Zuweisung eines Hintergrundprogramms zur Konsole

Format:

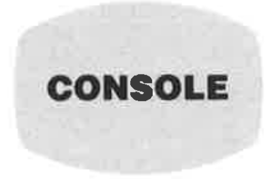
ATTACH programmname

Argument:

programmname — Der Name eines Programms, das von dieser Konsole gestartet und detached wurde.

Beschreibung:

Mit dem ATTACH-Programm wird ein mit ^D (detach; siehe Beispiel am Ende von Kapitel 2) in den Hintergrund geschaltetes Programm zur Konsole zurückgeschaltet.



- o CP/M-80
- o CP/M-Plus
- • MP/M-80
- o CP/M-86
- o CCP/M
- • MP/M-86

Ausgabe der Konsol-Nummer

Format:

CONSOLE

Beschreibung:

Der CONSOLE-Befehl zeigt die Nummer der Bediener-Konsole an. Sie liegt im Bereich von 0 bis 7 (bei MP/M-86 bis 254). Nach dem Kaltstart stimmen die Benutzer-Bereiche und die Konsol-Nummern überein (bei MP/M-86 nur bis Nummer 15). Die Benutzer-Bereiche lassen sich jedoch mit dem USER-Befehl umschalten.

Anwendung:

Die Feststellung der Konsol-Nummer ist u.U. nötig zur Benutzung des ABORT- und des STOPSPLR-Befehls.

Beispiel:

```
2A>CONSOLE↵
Console = 2
2A>USER 3↵
User Number = 3
3A>CONSOLE↵
Console = 2
3A>
```

(Das Beispiel zeigt, daß sich bei Änderung des Benutzerbereichs die Konsol-Nummer nicht ändert.)

- ○ CP/M-80
- ● CP/M-Plus
- ○ MP/M-80
- ○ CP/M-86
- ○ CCP/M
- ○ MP/M-86



Lesen und Setzen von Uhrzeit und Datum

Format:

1. DATE *c*
2. DATE *mm/dd/yy hh:mm:ss*
3. DATE *SET*

Argumente:

c	Wahlweise angebbares Argument, das die fortwährende Anzeige von Uhrzeit und Datum veranlaßt.
mm/dd/yy	Argument zum Setzen des Datums in der Reihenfolge: Monat (01...12), Tag (01...31) und Jahr (79...99).
hh:mm:ss	Argument zum Setzen der Uhrzeit in der Reihenfolge: Stunde (00...23), Minute (00...59) und Sekunde (00...59).
SET	Argument zum Setzen von Datum und Uhrzeit im Dialog- Modus.

Beschreibung:

DATE wird zum Lesen und Stellen der systemeigenen Softwareuhr verwendet. Diese Uhr wird z.B. gebraucht, um die Directory-Einträge mit Zeitinformationen zu versehen (time and date stamps).

Anwendung:

Wird nur DATE eingegeben, so wird Wochentag, Datum und Uhrzeit auf der Konsole ausgegeben. Mit dem Zusatz c läßt sich auf dem Bildschirm

eine laufende Digitaluhr darstellen (durch Eingabe irgendeines Zeichens abzubrechen). Zum Stellen der Uhr wird die gewünschte Zeit als Argument mit eingegeben. Um ein Sekunden-genaues Setzen zu gewährleisten, wartet DATE bis zur Eingabe eines weiteren Zeichens. Eine andere Möglichkeit ist die Angabe des Arguments SET. DATE fragt dann nach Datum und Uhrzeit, die auch einzeln gesetzt werden können.

Bei MP/M und CP/M-Plus wird zum Lesen und Stellen der Systemuhr der Befehl TOD benutzt.

Beispiel:

```
A>DATE↵
Sun 01/29/84 13:42:45
A>DATE 01/29/84 13:46:00↵
Strike key to set time
Sun 01/29/84 13:46:00
A>
```

- • CP/M-80
- o CP/M-Plus
- • MP/M-80
- o CP/M-86
- o CCP/M
- o MP/M-86



Symbolischer 8080-Debugger

Format:

DDT *dateibezeichnung*

Argument:

dateibezeichnung Datei, die mit dem DDT in den Speicher geladen werden soll.

Beschreibung:

DDT ist ein symbolischer Debugger, der es ermöglicht, Programme zu testen und Änderungen im Code vorzunehmen. Zur Darstellung können dabei entweder hexadezimale Ziffern oder Assemblercode dienen. Zum Programmtest können Hilfen wie Einzelschrittverarbeitung (trace) und Haltepunkte (breakpoints) in Anspruch genommen werden. Alle BDOS-Calls des zu testenden Programms werden in Echtzeit ausgeführt. Da sich der DDT selbst in den oberen Bereich der TPA (unter das BDOS) verlagert, können die üblichen CP/M-Befehlsdateien („.COM“) im richtigen Laufbereich (ab 100H) getestet werden.

Anwendung:

Wird DDT ohne Dateibezeichnung verwendet, so wird DDT nur allein geladen. Eine Datei kann jedoch auch nachgeladen werden. Wird beim DDT-Aufruf ein Dateiname mit angegeben, so wird diese Datei automatisch vom DDT ab Adresse 100H geladen. Eine hexadezimale Datei mit dem Dateityp „.HEX “wird dabei automatisch in Binärdaten gewandelt. Dateien (i.a. Programme), die mit DDT geändert wurden, können nach dem Warmstart mit dem SAVE-Befehl zurück auf die Platte geschrieben werden.

Bei den Systemen für die 86er-Prozessoren wird ein Debugger mit dem Namen „DDT86.CMD“ mitgeliefert.

Beispiel:

```
A>DDT PIP.COM↵
DDT VERS 2.2
NEXT PC
1E00 0100
_
```

NEXT gibt die erste Adresse hinter dem geladenen Programm an. Da das Programm ab 100H geladen wurde, läßt sich errechnen, daß es 29 Seiten lang ist (wichtig für SAVE-Befehl).

o CP/M-80	• CP/M-86
o CP/M-Plus	• CCP/M
o MP/M-80	• MP/M-86



Symbolischer 8086-Debugger

Format:

DDT86 *dateibezeichnung*

Argument:

dateibezeichnung Name einer Datei, i.a. eines Programms. Wird der Dateityp weggelassen, setzt DDT86 automatisch den Typ „.CMD“ ein.

Beschreibung:

DDT86 ist ein symbolischer Debugger, der es ermöglicht, Programme zu testen und Änderungen im Code vorzunehmen. Zur Darstellung können dabei entweder hexadezimale Ziffern oder Assemblercode dienen. Zum Programmtest können Hilfen wie Einzelschrittverarbeitung (trace) und Haltepunkte (breakpoints) in Anspruch genommen werden. Alle BDOS-Aufrufe des zu testenden Programms werden in Echtzeit ausgeführt.

Anwendung:

Wird DDT86 ohne Dateibezeichnung verwendet, so wird DDT86 nur allein geladen. Eine Datei kann jedoch auch nachgeladen werden. Wird beim DDT86-Aufruf ein Dateiname mit angegeben, so wird diese Datei automatisch von DDT86 in den Speicher geladen.

Zum Lieferumfang von CP/M-80 und MP/M-80 gehört der Debugger DDT.COM bzw. DDT.PRL.

Beispiel:

```
A>DDT86 MYPROG↵
DDT86 1.0
        START       END
CS   047D:0000   047D:002F
DS   0480:0000   0480:010F
_
```

(Die Datei MYPROG.CMD wird vom DDT86 in den Speicher geladen. START und END geben die Ladebereiche für das Code- und das Datensegment an.)

- CP/M-80
- CP/M-Plus
- MP/M-80
- CP/M-86
- CCP/M
- MP/M-86



Directory (Anzeige des Dateien-Inhaltsverzeichnisses)

Format:

1. DIR
2. DIR *d:*
3. DIR *filespec*
4. DIR *filespec, filespec* (nicht bei Standard CP/M)
5. DIR *filespec* [*opt*] (nicht bei Standard CP/M)

Argumente:

d:	Ein Laufwerk (A...P)
filespec	Dateiname oder Gruppenname unter Benutzung von [?] und [*] zur Einschränkung der angezeigten Dateien. DIR ohne Argument wird als DIR *.* interpretiert.
[*opt*]	Zusätzliche Informationen zur Steuerung besonderer Funktionen. Folgende Optionen sind bei MP/M und CCP/M implementiert:
[SYS]	Nur Dateien mit System-Attribut werden angezeigt.
[Gn]	Die Dateien für den Benutzerbereich n werden angezeigt.
	Folgende Optionen sind bei CP/M-Plus implementiert (nicht resident):
[ATT]	Zeigt die Datei-Attribute F1...F4 an.
[DATE]	Zeigt die Zeiteinträge für die Dateien an.
[DIR]	Zeigt nur Dateien ohne System-Attribut an.

[DRIVE=ALL]	Zeigt die Dateien aller dem System bekannten Laufwerke an.
[DRIVE=d]	Zeigt die Dateien des Laufwerks d an. Es können auch mehrere Laufwerke angegeben werden (durch Komma getrennt und in runde Klammern gesetzt).
[EXCLUDE]	Zeigt die Dateien an, die von der Dateigruppenbezeichnung ausgeschlossen werden.
[FF]	Sendet ein Formfeed-Zeichen zum Drucker, wenn das Drucker-Echo (^P) eingeschaltet ist.
[FULL]	Zeigt die Directory mit allen Informationen (Attribute, Paßwörter, Zeiteinträge) an.
[LENGTH=n]	Vereinbarung der Zeilenzahl pro Seite (n = 5...65536). Die Kopfzeile der Directory wird jede Seite wiederholt.
[MESSAGE]	Laufwerksnummer und Benutzerbereich, die gerade durchsucht werden, werden angezeigt.
[NOPAGE]	Die seitenweise Ausgabe wird abgeschaltet.
[NOSORT]	Die alphabetische Sortierung wird abgeschaltet.
[RO]	Es werden nur die Dateien mit Schreibschutz-Attribut angezeigt.
[RW]	Es werden nur die Dateien ohne Schreibschutz-Attribut angezeigt.
[SIZE]	Es wird jeweils der Dateiname und die Länge in KByte angezeigt.
[USER=ALL]	Es werden die Dateien aller Benutzerbereiche angezeigt.
[USER=n]	Es werden die Dateien des Benutzer-Bereichs n angezeigt. Es können auch mehrere Benutzer-Nummern angegeben werden (durch Komma getrennt und in runde Klammern gesetzt).

Beschreibung:

DIR listet das Inhaltsverzeichnis eines Laufwerks auf. Dabei werden nur Dateien des augenblicklichen Benutzerbereichs berücksichtigt. Dateien mit dem System-Attribut werden nicht angezeigt. Bei CP/M ist DIR ein residenter Befehl.

Beispiele:

```
A>DIR↵
A:DDT      COM : PIP    COM : ASM     COM : ED   COM
A:LOAD     COM : STAT   COM
A>DIR B:↵
B:DUMP     ASM : DUMP   HEX : DUMP    PRN
A>
```

Der folgende CP/M-Plus-Befehl listet alle Dateien vom Typ „.COM“ oder vom Typ „.OVR“ mit dem System-Attribut aller Benutzerbereiche der Laufwerke A und B auf.

```
1A>SDIR [USER=ALL,DRIVE=(A,B),SYS] *.COM *.OVR↵
```



o CP/M-80
o CP/M-Plus
• MP/M-80
o CP/M-86
• CCP/M
• MP/M-86

Rücksetzen der Plattenparameter

Format:

1. DSKRESET
2. DSKRESET *d:*
3. DSKRESET *d:,d:,d:...*

Argument:

d: Wahlweise ein oder mehrere Laufwerksbezeichnungen. Ohne Parameter gilt der Befehl für alle 16 möglichen Laufwerke.

Beschreibung:

Der DSKRESET-Befehl ermöglicht das Wechseln von Disketten bei Multi-Programming-Systemen. Der Befehl muß stets vor dem Wechsel gegeben werden. Wird ein Laufwerk, das zurückgesetzt werden soll, von einem anderen Programm noch benutzt (geöffnete Dateien), so gibt DSKRESET eine Fehlermeldung aus.

Beispiel:

```
0A>DSKRESET B:↵
Disk reset denied, Drive B: Console 2 Program PIP
0A>
```

(Die DSKRESET-Funktion ist nicht ausgeführt worden, da das von Konsole 2 gestartete PIP noch offene Dateien auf dem Laufwerk B hatte.)

- ● CP/M-80
- ● CP/M-Plus
- ● MP/M-80
- ○ CP/M-86
- ○ CCP/M
- ○ MP/M-86





Hexadezimale Dateilistung

Format:

DUMP dateibezeichnung

Argument:

Dateibezeichnung mit Dateiname und Dateityp.

Beschreibung:

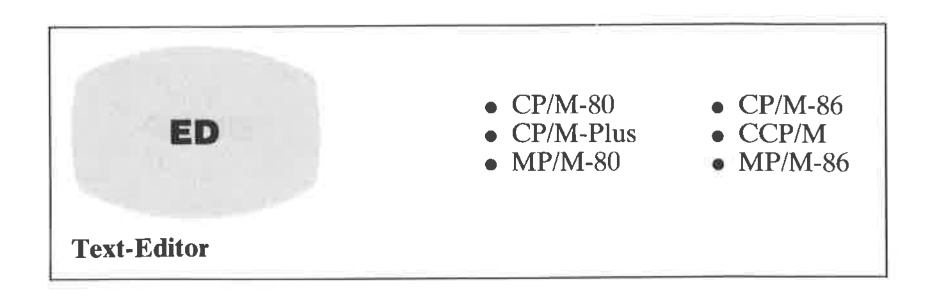
DUMP zeigt in hexadezimaler Darstellung den Inhalt einer beliebigen Disketten- oder Plattendatei auf dem Bildschirm an. Hierbei werden in jeder Zeile links die absolute Adresse ab Dateianfang und rechts sechzehn Bytes angezeigt. Bei der CP/M-Plus-Version wird außerdem noch der Inhalt in ASCII-Zeichen mit angezeigt.

Anwendung:

Das DUMP-Programm wird von Digital Research sowohl als „.COM"-Datei als auch als Assembler-Quellcode („.ASM"-Datei) mitgeliefert. Es eignet sich aufgrund seiner Einfachheit besonders zur Einarbeitung in die Assemblersprache.

Beispiel:

```
A>DUMP PROGRAM.COM↵
0000: C3 20 01 41 42 74 EC 09 54 F7 40 93 66 14 5F F2
0010: DB 04 CD 33 09 D3 05 21 53 10 CD 30 10 DB 04 C9
0020: 31 00 40 11 4F 01 0E 09 CD 00 05 01 66 02 CD 22
...
```



Format:

1. ED dateibezeichnung
2. ED dateibezeichnung *d:*

Argumente:

dateibezeichnung — Bezeichnung der zu editierenden „Textdatei." Die Datei darf nicht den Typ „.BAK" oder „.$$$" haben.

d: — Optionale Laufwerksangabe für den Fall, daß die zu editierende Datei zu einem anderen Laufwerk geschickt werden soll als zu dem, von dem die Originaldatei stammt.

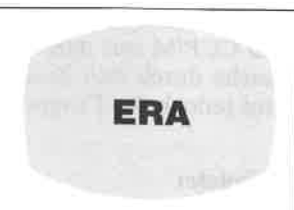
Beschreibung:

ED ist ein zeilenorientierter Texteditor (im Gegensatz zu bildschirmorientierten Editoren wie z.B. WordStar). Alle Edit-Funktionen werden in einem Puffer im RAM durchgeführt. Der Text muß von der Datei in den Puffer geladen und nach Beendigung der Bearbeitung zur Datei zurückgeschrieben werden. ED löscht die Original-Datei nicht, sondern behält sie als sogenannte „Backup"-Datei mit dem Dateityp „.BAK".

Anwendung:

Details über die Anwendung von ED befinden sich in Kapitel 4.

- CP/M-80
- CP/M-Plus
- MP/M-80
- CP/M-86
- CCP/M
- MP/M-86



Löschen von Plattendateien

Format:

ERA dateibezeichnung
ERA dateigruppe

Argumente:

dateibezeichnung	Vollständiger Name (ggf. mit Laufwerksangabe) der zu löschenden Datei.
dateigruppe	Zum Löschen mehrerer oder aller Dateien eines Laufwerks können die Platzhalterzeichen ? und * verwendet werden.

Beschreibung:

ERA wirkt nur auf die Dateien des augenblicklichen Benutzerbereichs. Dateien mit dem Schreibschutz-Attribut werden nicht gelöscht. Bei CP/M-80 führt ein Löschversuch bei einer schreibgeschützen Datei zur BDOS-Fehlermeldung mit Abbruch des Befehls und folgendem Warmstart. Wird ein ERA-Befehl zum Löschen aller Dateien (*.*) eingegeben, so muß der Benutzer noch eine Bestätigung geben (siehe Beispiel). Bei CP/M-Plus wird in jedem Fall bei Verwendung von ? oder * eine Bestätigung verlangt.

Anwendung:

ERA ist beim Standard-CP/M ein residenter Befehl. Bei MP/M und CCP/M muß es als Datei (ERA.PRL bzw. ERA.CMD) auf der Platte stehen. Directory-Einträge, die nicht den CP/M-Konventionen entsprechen, können u.U. mit ERA nicht gelöscht werden. In solchen Fällen hilft oft nur das Umkopieren aller noch verwendbaren Dateien und Neuformatieren der Diskette.

Das Löschen von Dateien mit Bestätigung ist bei den Systemen MP/M und CCP/M mit dem Befehl ERAQ möglich. Bei CP/M-Plus kann das gleiche durch den Zusatz [CONFIRM] erreicht werden. In diesem Fall wird jedoch das Programm ERASE.COM benötigt.

Beispiele:

```
A>ERA BEISPIEL.TXT↵
```

(Dieser Befehl löscht die Datei BEISPIEL.TXT in Laufwerk A.)

```
A>ERA *.*↵
ALL (Y/N)? Y
A>
```

(CP/M verlangt Bestätigung, bevor es alle Dateien löscht.)

o CP/M-80	o CP/M-86
o CP/M-Plus	• CCP/M
• MP/M-80	• MP/M-86



Löschen von Dateien mit Quittung

Format:

ERAQ dateigruppenbezeichnung

Argument:

dateigruppenbezeichnung	Mit der Dateigruppenbezeichnung wird dem ERAQ-Befehl eine Gruppe von Dateien mitgeteilt, die in ASCII-Reihenfolge zu löschen sind.

Beschreibung:

ERAQ wird eingesetzt zum Löschen mehrerer Dateien in einem Arbeitsgang. Bei jeder Datei wird vor dem Löschen eine Bestätigung verlangt.

Anwendung:

ERAQ läßt sich besonders gut zur Bereinigung von Directories verwenden. Ein besonderer Vorteil von ERAQ ist es auch, daß man hiermit Dateien löschen kann, deren Namen kleine Buchstaben oder Zeichen enthalten, die vom Befehls-Interpreter nicht angenommen werden. (Siehe auch den Befehl ERA.)

Beispiel:

```
1A>ERAQ C:*.PRL↵
C:ASM       PRL ?Y
C:CONSOLE   PRL ?N
C:DIR       PRL ?Y
C:DSKRESET  PRL ?Y
1A>
```

GENCMD

o CP/M-80	• CP/M-86
o CP/M-Plus	• CCP/M
o MP/M-80	• MP/M-86

Erzeugung einer Befehlsdatei aus einer Intel-Hex-Datei

Format:

GENCMD dateiname [parameter]

Argumente:

dateiname — Name einer vorhandenen Datei vom Typ „.H86" (Typ wird nicht mit eingegeben). Die erzeugte Befehlsdatei trägt den gleichen Namen, der Typ ist jedoch „.CMD".

parameter — Optionale Parameter zur genaueren Spezifizierung der Zieldatei. Folgende Schlüsselwörter sind erlaubt:

- 8080 — 8080-Speichermodell
- CODE — Es folgen Adreßangaben für den Programmcode
- DATA — Es folgen Adreßangaben für den Datenbereich
- STACK — Es folgen Adreßangaben für den Stackbereich
- EXTRA — Es folgen Adreßangaben für das Extrasegment
- X1..X4 — Es folgen Adreßangaben für Zusatzsegmente

Die Adreßangaben werden in eckige Klammern gesetzt. Die Adressen werden in Paragraphen (das sind 16-Byte-Einheiten) angegeben. Ein Buchstabe vor der Adresse vereinbart ihre Bedeutung:

- A — Es folgt die absolute Ladeadresse der Gruppe
- B — Es folgt die Gruppenadresse innerhalb der Hex-Datei
- M — Es folgt der minimale Speicherbedarf
- X — Es folgt der maximale Speicherbedarf

Beschreibung:

Der GENCMD-Befehl dient zur Erzeugung von Befehlsdateien für die 86er-Prozessorfamilie. Die Eingangsdatei muß im erweiterten Intel-Hex-Format sein (Dateityp „.H86“). Die Befehlsdatei vom Typ „.CMD“ enthält am Anfang einen sogenannten Header, der dem Programmlader die benötigten Informationen gibt.

Anwendung:

GENCMD wird i.a. in Verbindung mit dem Assembler ASM86 eingesetzt. Zur Anwendung des GENCMD-Befehls muß der Benutzer erst einmal Informationen über die Speicheraufteilung seines Rechners haben, um das richtige Speichermodell auswählen zu können. Am einfachsten ist das 8080-Speichermodell; hier liegen Code und Daten im gleichen Bereich. Genauere Informationen finden Sie in den Handbüchern von Digital Research.

Beispiel:

```
0A>GENCMD BEISPIEL CODE[A40] DATA[M30,XFFF]↵
```

Es wird aus der Datei BEISPIEL.H86 die Datei BEISPIEL.CMD erzeugt. Der Code wird zur Adresse 400H gelegt. Der Datenbereich braucht mindestens 300H Bytes, kann vom Programm aber bis zu einem Bereich von FFF0H Bytes ausgedehnt werden.



- o CP/M-80
- o CP/M-Plus
- ● MP/M-80
- o CP/M-86
- o CCP/M
- o MP/M-86

Erzeugung einer Datei im Intel-Hex-Format

Format:

GENHEX programmname.COM offset

Argumente:

programmname	Der Name des zu transformierenden Programms (vom Dateityp „.COM"). Dem Programmnamen kann ein Laufwerkskennzeichen, das dann auch für die Zieldatei gilt, vorangesetzt werden.
offset	Hexadezimaler Wert zur Vereinbarung eines Adreß-Offsets.

Beschreibung:

Dieser Befehl generiert aus einer Datei vom Typ „.COM" eine Datei vom Typ „.HEX" und addiert den durch das Argument „offset" angegebenen Wert zur Adresse.

Beispiel:

```
1A>GENHEX ACTION.COM 100↵
```

- o CP/M-80
- • CP/M-Plus
- o MP/M-80
- o CP/M-86
- • CCP/M
- o MP/M-86



Verwaltung einer Manual-Datei

Format:

1. HELP
2. HELP *bezeichnung [option]*
3. HELP *bezeichnung unterbez1 unterbez2 ... [option]*
4. HELP *[EXTRACT]*
5. HELP *[CREATE]*

Argumente:

bezeichnung	Name des zu beschreibenden Begriffs.
unterbez1 bis unterbez8	Name eines untergeordneten Begriffs zur genaueren Spezifizierung des Textteils.
[option]	Option für die Steuerung des Ausgabeformats. Implementiert sind:
[NOPAGE] bzw. [N]	Die seitenweise Ausgabe bei CP/M-Plus wird ausgeschaltet.
[P]	Die Ausgabe geschieht seitenweise (CCP/M).

Beschreibung:

Das HELP-Programm verwaltet eine Auskunftsdatei (Manual-Datei HELP.HLP), die Textinformationen über Befehle, Programme usw. des Systems enthält. Der Zugriff auf einzelne Kapitel des Manuals geschieht über symbolische Namen. Jedes Kapitel kann in Unterkapitel unterteilt werden, auf die wiederum über symbolische Namen zugegriffen werden kann (bis zu einer Schachtelungstiefe von 8).

Anwendung:

Format 1 erzeugt eine Inhaltsangabe des Manuals und wartet dann auf weitere Eingaben. Mit Format 2 und 3 läßt sich über Stichworte direkt auf

bestimmte Abschnitte des Manuals zugreifen. Format 4 und 5 dienen zur Änderung des Manuals: [EXTRACT] bildet eine Textdatei (HELP. DAT) aus der Manual-Datei. Diese kann dann mit einem Editor bearbeitet werden. Mit [CREATE] wird aus dieser Textdatei dann wieder eine Manual-Datei (HELP.HLP) gebildet.

Beispiele:

```
A>HELP ED↵
```

(Eine allgemeine Beschreibung des Editors ED wird gegeben.)

```
A>HELP ED COMMANDS↵
```

(Eine Beschreibung der ED-Befehle wird ausgegeben.)

- o CP/M-80
- • CP/M-PLUS
- o MP/M-80
- o CP/M-86
- o CCP/M
- o MP/M-86



Initialisieren der Directory

Format:

INITDIR d:

Argument:

d: Laufwerksangabe (A...P)

Beschreibung:

Für die erweiterten Directory-Einträge (XFCB extended file control block) sind bei CP/M-Plus bestimmte Stellen im Directory-Bereich reserviert. Das INITDIR-Programm bereitet den Directory-Bereich hierfür vor. Ist die Platte bereits initialisiert, so kann man mit INITDIR sämtliche XFCBs löschen und den Bereich wieder freigeben.

Das Directory-System von CP/M-Plus ist eine Weiterentwicklung des bei MP/M bestehenden Verfahrens. Paßwort- und Zeiteinträge sind nicht verträglich.

Beispiele:

```
A>INITDIR B:↵
INITDIR WILL ACTIVATE TIME STAMPS FOR SPECIFIED DRIVE.
Do you want re-format the directory on drive: B  (Y/N)? Y↵
```

```
A>INITDIR B:↵

Directory already re-formatted.
Do you want to recover time/date directory space  (Y/N)? Y↵

A>
```

(Im ersten Beispiel wurde die Directory des Laufwerks B für die erweiterten Einträge initialisiert. Im zweiten Beispiel wurde der Directory-Platz wieder freigegeben.)

- o CP/M-80
- • CP/M-Plus
- • MP/M-80
- o CP/M-86
- o CCP/M
- o MP/M-86



Verbinden von speicherrelativen Programmen (linking)

Format:

1. LINK programm1[opt],programm2[opt],....
2. LINK zielprogramm[opt]=programm1[opt],programm2[opt],...

Argumente:

programm1... Name einer Datei im REL-Format. Wird kein Dateityp angegeben, so sucht LINK nach Dateien vom Typ „.IRL“ und „.REL“. Wird kein Zielprogramm angegeben, so erhält die Objekt-Datei und die Symboltabelle den gleichen Namen wie die als erstes angegebene REL-Datei.

zielprogramm Dateiname für das lauffähige Programm und für die Symboltabelle (vom Typ „.SYM“). Wird kein Dateityp angegeben, so erhält die Objekt-Datei den Typ „.COM“ oder, wenn als Option vereinbart, den Typ „.PRL“, „.SPR“ oder „.RSP“.

opt Eine oder mehrere durch Komma getrennte Linker-Anweisungen. Folgende Optionen sind implementiert:

A Additional Memory (zusätzlicher Speicher). Der Linker benutzt zur Speicherung von Tabellen Puffer-Dateien auf der Platte.

B Objekt-Datei ist eine spezielle Form von „.SPR“-Datei, die vom GENCPM-Programm benötigt wird (nur bei CP/M-Plus).

Dnnnn Data Origin. Der DATA-Bereich soll bei Adresse nnnn (Hex) beginnen.

G<label>	Go. Die Startadresse für das Programm wird auf „label“ gelegt. „label“ muß im Programm als ENTRY bzw. PUBLIC deklariert sein.
Lnnnn	Für nicht unter CP/M laufende Programme kann die Ladeadresse statt auf 100H auf den Wert nnnn gelegt werden.
Mnnnn	Ein zusätzlicher Speicherbereich für den Header von „.PRL“-Dateien wird vereinbart. Diese Angabe wird vom MP/M-Programm-Lader verarbeitet.
NL	No List. Die Symbol-Tabelle wird nicht auf der Konsole ausgegeben.
NR	No Record. Es wird keine Datei mit der Symbol-Tabelle erzeugt.
OC	Output COM. Die Objekt-Datei ist vom Typ „.COM“ (Dies ist der Normalfall).
OP	Output PRL. Die Objekt-Datei ist vom Typ „.PRL“.
OR	Output RSP. Die Objekt-Datei ist vom Typ „.RSP“.
OS	Output SPR. Die Objekt-Datei ist vom Typ „.SPR“.
Pnnnn	Program Origin. Die Basisadresse für den Programm-Code ist nnnn (Hex).
Q	Die Symbole mit einem Fragezeichen als erstes Zeichen werden mit in die Symboltabelle aufgenommen.
S	Search. Es werden nur die Teile der vorstehenden Datei geladen, die benötigt werden. Die S-Option wird bei Benutzung von Library-Dateien angewendet.
$CY	Die Konsol-Ausgabe wird zum Drucker geleitet.
$CZ	Die Konsol-Ausgabe wird unterdrückt.
$Id	Die Zwischendateien werden auf Laufwerk d aufgebaut.
$Ld	Die Library-Dateien (Bibliotheken) werden auf Laufwerk d gesucht.
$Od	Die Objekt-Datei wird auf Laufwerk d geschrieben.
$Sd	Die Symbol-Datei wird auf Laufwerk d geschrieben.

Beschreibung:

LINK ist ein Programm zum Verbinden (link) und Verschieben (relocate) von Dateien im Microsoft-REL-Format. Dieses Format wird von zahlreichen Assemblern (z.B. RMAC von Digital-Research oder M80 von Microsoft) und Compilern (z.B. BASCOM von Microsoft) verwendet. Auch das CP/M-Plus-Betriebssystem wird in diesem Format geliefert, da hierdurch eine einfache Möglichkeit zur Verbindung mit den Hardware-Treibern besteht. LINK bietet die Möglichkeit, Programme mit Overlay-Technik zu verarbeiten. Nähere Informationen sind in den entsprechenden Manualen von Digital Research zu finden.

Anwendung:

LINK wird nach dem Assemblieren oder Kompilieren eines Programms benutzt. Kommt man mit einer Eingabezeile nicht aus, so kann man als letztes Zeichen der Zeile ein &-Symbol setzen; LINK wartet dann auf weitere Zeilen. Durch Angabe der A-Option lassen sich fast beliebig lange Programme linken. Die Verarbeitungszeit wird dadurch jedoch erheblich verlängert.

Beispiele:

```
A>LINK BNKBIOS3[B]=MYBIOS,DISKS,CONSOLE,UHR,SCB↵
```

(LINK erzeugt aus den Dateien MYBIOS.REL, DISKS.REL, CONSOLE.REL, UHR.REL und SBC.REL die Objekt-Datei BNKBIOS3.SPR und die Symbol-Datei BNKBIOS3.SYM. BNKBIOS3.SPR wird vom CP/M-Programm GENCPM benötigt.)

```
A>LINK MYPROG↵
```

(LINK erzeugt die Dateien MYPROG.COM und MYPROG.SYM aus der Datei MYPROG.REL.)



- ● CP/M-80
- o CP/M-Plus
- ● MP/M-80
- o CP/M-86
- o CCP/M
- o MP/M-86

Bildung einer Befehls-Datei aus einer Hex-Datei

Format:

LOAD dateiname

Argument:

dateiname — Name einer bestehenden Datei vom Typ „.HEX“. Die erzeugte Befehlsdatei erhält den gleichen Namen mit dem Typ „.COM“.

Beschreibung:

Das LOAD-Programm lädt eine Datei im Intel-Hex-Format und macht daraus ein ausführbares Programm vom Typ „.COM“.

Anwendung:

LOAD wird i.a. im Zusammenhang mit dem Assembler ASM zur Herstellung von CP/M-Programmen benutzt.

Beispiel:

```
A>ASM BEISPIEL↵

A>LOAD BEISPIEL↵

A>BEISPIEL↵
```

Der Assembler erzeugt eine Datei BEISPIEL.HEX, und LOAD bildet daraus die Datei BEISPIEL.COM, die dann als Befehl aufgerufen werden kann.

- ● CP/M-80
- o CP/M-Plus
- o MP/M-80
- o CP/M-86
- o CCP/M
- o MP/M-86



Rekonfiguration des CP/M-Systems

Format:

MOVCPM bb *

Argumente:

bb	Speichergröße des Systems in KBytes. Wird für bb ein Stern eingegeben, so ermittelt MOVCPM die Speichergröße selbst.
*	Wird als zweites Argument ein Stern angegeben (optional), so bleibt das neue System in der TPA und kann mit einem nachfolgenden SAVE oder SYSGEN-Befehl auf die Diskette geschrieben werden. Wird der Stern weggelassen, so kopiert MOVCPM das System in den Laufbereich und startet es.

Beschreibung:

Das MOVCPM-Programm gehört zum Lieferumfang des Standard-CP/M und wird eingesetzt bei einer Änderung der System-Speichergröße. MOVCPM enthält als Bestandteil ein komplettes CP/M; das BIOS ist die Version für das Intel-Entwicklungssystem MDS-800. Deshalb kann das MOVCPM-Programm auf anderen Rechnern nur bedingt eingesetzt werden. Viele Rechnerhersteller liefern jedoch eigene Programme zum Generieren und Kopieren des Betriebssystems mit.

Anwendung:

Das GENCPM-Programm kann nur in einem CP/M-80-System mit der gleichen Lizenznummer benutzt werden. Anwendungsbeispiele finden Sie in Kapitel 5.



- ○ CP/M-80
- ○ CP/M-Plus
- ● MP/M-80
- ○ CP/M-86
- ○ CCP/M
- ● MP/M-86

Laden des MP/M-Betriebssystems

Format:

MPMLDR

Beschreibung:

MPMLDR lädt die Datei MPM.SYS in den Laufbereich, gibt eine Liste von Ladeinformationen auf die Konsole aus und startet das MP/M-Betriebssystem. Das MPMLDR-Programm ist vollständig autark und muß ein auf den Rechner angepaßtes BIOS enthalten.

Anwendung:

Der MP/M-Lader kann sowohl als CP/M-Programm von einem laufenden CP/M-System aus gestartet oder aber auf die Systemspuren der Platte gebracht werden. In diesem Fall wird er dann vom Bootstrap-Lader in den Speicher geladen.

Beispiel:

```
A>MPMLDR↵

MP/M-II V2.1 Loader
Copyright © 1981, Digital Research

Nmb of Consoles = 2
Breakpoint RST # = 6
Z80 Alternate register set saved/restored by dispatcher

Memory Segment Table
SYSTEM  DAT  FF00H  0100H
...
...

0A>
```

○ CP/M-80
○ CP/M-Plus
● MP/M-80

○ CP/M-86
○ CCP/M
● MP/M-86

MPMSTAT

Anzeige des MP/M-System-Status

Format:

MPMSTAT

Beschreibung:

MPMSTAT gibt eine Liste aller Multi-Task-spezifischen Vorgänge im Betriebssystem aus. Die Liste enthält folgende Informationen:

- Statusinformationen einschließlich der Anzahl der Konsolen
- Prozesse, die bereit zur Ausführung sind
- Prozesse, die auf eine Queue-Nachricht warten
- Prozesse, die auf das Freiwerden eines Queuebuffers warten
- Prozesse, die für eine bestimmte Zeit warten
- Prozesse, die im Abfragemodus auf ein externes Ereignis warten
- Prozesse, die auf einen Interrupt warten
- Eine Liste aller geöffneten Queues
- Konsol-Nummern mit den belegenden Prozessen
- Konsol-Nummern mit den Prozessen, die auf ein Freiwerden warten
- Drucker-Nummern mit den belegenden Prozessen
- Drucker-Nummern mit den wartenden Prozessen
- Liste der laufenden Programme und ihrer Speichersegmente
- Angabe der Konsolen, von denen die laufenden Programme gestartet wurden

Anwendung:

MPMSTAT kann z.B. eingesetzt werden, um festzustellen, ob ein Programm (z.B. durch einen besetzten Drucker) blockiert wird, oder (vor Benutzung des ABORT-Befehls) von welcher Konsole aus es gestartet wurde. Da MPMSTAT mehr als 24 Zeilen beschreibt, empfiehlt sich die Verwendung von ^S und ^Q.

PIP

- CP/M-80
- CP/M-Plus
- MP/M-80
- CP/M-86
- CCP/M
- MP/M-86

Allgemeines Kopierprogramm (Peripheral Interchange Program)

Format:

1. PIP { *d*:zieldatei[gn] | d: } =*d*:quelldatei[o]
2. PIP { dev: | *d*:dateibez } = { dev: | *d*:dateibez } [o], { dev: | *d*:dateibez } ,...
3. PIP d:[gn]=*d*:dateigruppe[o]
4. PIP *d*:dateigruppe[gn]=d:[o]

Argumente:

{*d*:zieldatei} — Bei Format 1 kann der Benutzer durch optionale Angabe einer Zieldatei für die Kopie einen anderen Namen wählen als den der Quelldatei.

d:quelldatei — Der Name der Quelldatei muß immer angegeben werden.

[gn] — (Nicht bei CP/M-80) optionale Angabe eines Benutzerbereichs mit der Nummer n für die Zieldatei.

[o] — Hinter jeder Quellenbezeichnung können Optionen angegeben werden.

dev: — Im Format 2 muß vom Benutzer entweder für „dev:“ d:dateibez. die Kurzbezeichnung der anzusprechenden Geräteeinheit (z.B. CON:) oder eine Dateibezeichnung (wahlweise mit Laufwerksbezeichnung) angegeben werden.

Formate 3 und 4 — Die Formate 3 und 4 werden zum Kopieren von Dateigruppen auf ein anderes Laufwerk benutzt. Die Laufwerksbezeichnungen müssen verschieden sein.

Eine Laufwerksbezeichnung kann wegfallen; es wird dafür das Arbeitslaufwerk angenommen.

Beschreibung:

PIP (Peripheral Interchange Program) ist ein universelles Programm zur Übertragung von Daten zwischen Plattendateien und zwischen Peripheriegeräten. Es bietet zahlreiche Funktionen für die Formatumwandlung der Daten und andere spezielle Aufgaben, die durch Optionen (in eckigen Klammern hinter dem Kopierziel) vereinbart werden. Eine genaue Beschreibung von PIP mit einer Liste der Optionen finden Sie in Anhang H.

Anwendung:

PIP-Operationen können durch Aufruf von PIP mit gleichzeitiger Parameterangabe durchgeführt werden. Es besteht jedoch auch die Möglichlichkeit, PIP allein aufzurufen, wonach es sich mit einem Stern meldet. Jetzt können mehrere Operationen ausgeführt werden, ohne daß dabei das PIP-Programm jeweils neu geladen werden muß. Nach jeder Operation meldet es sich wieder mit dem Stern. Zur Beendigung wird dann eine Leerzeile (nur CR) eingegeben (Abbruch mit ^C ist auch möglich).

Beispiele:

```
A>PIP B:=*.*↵
```

Dieser Befehl kopiert alle Nicht-System-Dateien des aktuellen Laufwerks (darf nicht B sein) zum Laufwerk B.

```
A>PIP↵
*DATEI2=TEST2↵
*LST:=DATEI2↵
*↵
A>
```

Mit dieser Befehlsfolge wird PIP aufgerufen, eine Datei DATEI2 mit dem Inhalt von TEST2 erzeugt und der Inhalt von DATEI2 auf den Drukker ausgegeben.

```
A>PIP LST:=TEXT2[FPZ]↵
```

Die Datei TEXT2 wird auf den Drucker ausgegeben. Dabei werden alle Formfeed-Zeichen gelöscht (F-Option) und alle 60 Zeilen neue Formfeeds eingefügt (P-Option). Da die Datei mit einem Textverarbeitungsprogramm erstellt wurde, das das Paritätsbit der ASCII-Dateien für interne Zwecke verwendet (z.B. WordStar im „Document mode"), wird es vor der Ausgabe auf 0 gesetzt (Z-Option).

```
A>PIP A:=B:*.*[V]↵
```

Alle Dateien von Laufwerk B (im augenblicklichen Benutzerbereich) werden, auch wenn sie das System-Attribut haben (R-Option), zum Laufwerk A kopiert. Jede kopierte Datei wird daraufhin überprüft, ob sie fehlerfrei auf die A geschrieben wurde (V-Option).

- o CP/M-80
- o CP/M-Plus
- • MP/M-80
- o CP/M-86
- • CCP/M
- • MP/M-86

PRINTER

Anwahl eines Druckers

Format:

1. PRINTER
2. PRINTER n

Argument:

n Nummer des Druckers (0,1,...)

Beschreibung:

Mit dem PRINTER-Befehl läßt sich der eingebenden Konsole ein bestimmter Drucker zuordnen (Format 2) bzw. abfragen, welcher Drukker gerade zugeordnet ist. Nach dem Kaltstart ist für alle Konsolen der Drucker 0 zugeordnet.

Anwendung:

Vor dem Start eines Programms, das eine Drucker-Ausgabe enthält, läßt sich mit PRINTER der physikalische Drucker (Plotter o.ä.) zuordnen. Ein Drucker kann nicht von mehreren Programmen gleichzeitig benutzt werden.

Beispiele:

```
0A>PRINTER↵
Printer Number = 0

0A>PRINTER 1↵
Printer Number = 1

0A>
```

(Im ersten Beispiel wurde der angewählte Drucker festgestellt und im zweiten Beispiel der Drucker 1 angewählt.)

PRLCOM

- o CP/M-80
- o CP/M-Plus
- ● MP/M-80
- o CP/M-86
- o CCP/M
- o MP/M-86

Erzeugung einer „.COM"-Datei aus einer „.PRL"-Datei

Format:

PRLCOM programmname1 *programmname2*

Argumente:

programmname1: Name einer Datei im PRL-Format. Wird der Dateityp weggelassen, wird automatisch „.PRL" angenommen.

programmname2 Optionaler Name der Zieldatei (ausführbares CP/M-Programm). Wird der Dateityp weggelassen, wird „.COM" eingesetzt. Ohne Angabe der Zieldatei wird programmname1 eingesetzt.

Beschreibung:

Dieser Befehl erzeugt aus einem verschiebbaren „.PRL"-Programm ein CP/M-verträgliches „.COM"-Programm, das absolut geladen wird.

Anwendung:

Der PRLCOM-Befehl kann benutzt werden, um MP/M-Programme unter CP/M lauffähig zu machen (Es dürfen dann jedoch keine speziellen MP/M-Systemaufrufe im Programm sein !). Die erzeugten „.COM"-Programme (die etwas kürzer werden) sind auch weiterhin unter MP/M lauffähig, jedoch nur in absoluten Speichersegmenten.

Beispiel:

```
1A>PRLCOM ERAQ↵
```

(Aus dem Programm ERAQ.PRL wird ein Programm ERAQ.COM generiert.)

- CP/M-80
- CP/M-Plus
- MP/M-80
- CP/M-86
- CCP/M
- MP/M-86

REN

Umbenennen (Rename) einer Datei

Format:

REN neu = alt

Argumente:

neu	Dateiname einer noch nicht existierenden Datei.
alt	Dateiname einer existierenden Datei, die umbenannt werden soll. Wird das Laufwerk mit angegeben, darf es sich nicht von dem bei „neu" angegebenen Laufwerk unterscheiden.

Bei CP/M-Plus, MP/M und CCP/M dürfen für neu und alt auch Dateigruppenbezeichnungen eingesetzt werden. Die Lage der eingefügten Joker-Zeichen ? und * muß bei „neu" und „alt" exakt übereinstimmen, während die anderen Zeichen Unterschiede aufweisen müssen.

Beschreibung:

Der REN-Befehl dient zur Umbenennung von Plattendateien. Es wird dabei keine Kopie angelegt, sondern lediglich der Namenseintrag in der Directory geändert.

Anwendung:

Beim CP/M ist REN ein residenter Befehl, bei den anderen Systemen muß es als Befehlsdatei auf der Platte stehen. Bei CP/M-Plus ist der Grundbefehl resident; bei Angabe von Dateigruppenbezeichnungen wird das Programm RENAME.COM von der Platte nachgeladen. Die Angabe von Dateigruppenbezeichnungen erlaubt die Umbenennung mehrerer Dateien mit ähnlichen Namen.

Beispiele:

```
A>REN TEXTNEU.HEX = TEXTALT.HEX↵
```

Dieser Befehl benennt die Datei TEXTALT.HEX in TEXTNEU.HEX um.

```
1A>REN *.PAS = *.SRC↵
SIEVE    .PAS=SIEVE    .SRC
NEWTON   .PAS=NEWTON   .SRC
TEST     .PAS=TEST     .SRC
1A>
```

Es wurden durch einen Befehl alle Dateien vom Typ „.SRC“ in Dateien vom Typ „.PAS“ umgewandelt.

o CP/M-80	o CP/M-86	**RMAC**
• CP/M-Plus	o CCP/M	
• MP/M-80	o MP/M-86	
		8080 Makro-Assembler

Format:

RMAC dateiname *option*

Argumente:

dateiname — Name einer Quelldatei. Wird der Dateityp weggelassen, so wird nach einer Datei vom Typ „.ASM" gesucht. Die vom Assembler erzeugten Dateien erhalten denselben Namen.

option — Optional ein oder mehrere Steuerparameter, beginnend mit einem $-Zeichen:

- *Rd* — Laufwerk für „.REL"-Datei (A..O, Z)
- *Sd* — Laufwerk für „.SYM"-Datei (A..O, X, P, Z)
- *Pd* — Laufwerk für .PRN-Datei (A..O, X, P, Z)

Dabei sind A..O Laufwerksbezeichnungen; X bedeutet Konsolausgabe, Y bedeutet Druckerausgabe, und Z unterdrückt die Ausgabe.

Beschreibung:

RMAC ist ein komfortabler 8080-Makro-Assembler, der verschiebbaren Objekt-Code im REL-Format erzeugt. Dieses Format gestattet die getrennte Behandlung von Programm- und Datenbereichen und das Zusammenbinden mit anderen, getrennt assemblierten Programmteilen. Die Konvertierung in lauffähigen Programmcode wird von einem Linker vorgenommen (siehe LINK).

Anwendung:

Damit getrennt assemblierte Programmteile später miteinander verkehren können, muß es gemeinsame Symbole (Publics) geben, die dem Linker mitgeteilt werden. Der Linker setzt dafür die tatsächlichen Adressen ein.

Beispiel:

```
A>RMAC TEST $PX SB RB↵
```

In diesem Beispiel wird die Datei TEST.ASM von Laufwerk A assembliert, das Listing zur Konsole geschickt und die Symbol-Tabelle (TEST. SYM) sowie die Objekt-Datei (TEST.REL) zum Laufwerk B geschrieben.

- ● CP/M-80
- ● CP/M-Plus
- o MP/M-80
- o CP/M-86
- o CCP/M
- o MP/M-86

SAVE

Sichern des TPA-Bereichs als Plattendatei

Format:

SAVE p dateibez (bei CP/M-80)

SAVE (bei CP/M Plus)

Argumente:

p	Anzahl der zu speichernden Seiten (je 256 Bytes) im Bereich 1...255.
dateibez	Gewünschte vollständige Dateibezeichnung der zu erstellenden Datei.

Beschreibung:

Der SAVE-Befehl dient zum Sichern des TPA-Bereichs unter CP/M. Bei CP/M-80 ist der Befehl resident und wird gegeben, nachdem die zu sichernden Daten in der TPA stehen (z.B. nach Benutzung des DDT). Gesichert werden dann die mit p angegebenen Seiten ab Adresse 100H.

Bei CP/M-Plus wird das SAVE-Programm aufgerufen, bevor die Daten in den TPA gebracht werden (der CCP zerstört unter CP/M-Plus den TPA-Bereich !). Eine spezielle Routine wird von SAVE zum oberen Ende der TPA kopiert, wo sie dann beim nächsten Warmstart (vor dem Laden des CCP) aktiv wird (siehe Beispiel).

Anwendung:

Der SAVE-Befehl wird hauptsächlich im Zusammenhang mit dem Debugger (z.B. DDT oder SID) verwendet: Eine Datei wird mit dem Debugger in die TPA geladen, dort geändert und dann mit SAVE auf die Platte zurückgeschrieben. Einige Debugger besitzen selbst einen Befehl zum Sichern der TPA, so daß die Benutzung von SAVE entfallen kann.

Beispiele:

```
A>DDT PIP.COM↵
DDT VERS 2.2
NEXT      PC
1E00     0100
–
G0

A>USER 1↵
A>SAVE 29 PIP.COM↵
A>
```

(Übertragung des PIP-Programms in einen anderen Benutzerbereich unter CP/M-80.)

```
A>SID DUMP.COM↵
```

(Bearbeitung des Programms mit SID)

```
#G0↵
```

(Abbruch von SID)

```
SAVE Ver 3.0
File (or RETURN to exit)?DUMP2.COM↵
From? 100↵
To? 400↵

A>
```

(Sichern der geänderten Version vom DUMP.COM unter CP/M-Plus.)



- o CP/M-80
- o CP/M-Plus
- ● MP/M-80
- o CP/M-86
- o CCP/M
- o MP/M-86

Starten eines Programms zum festgesetzten Zeitpunkt (Schedule)

Format:

SCHED mm/dd/yy hh:mm befehlszeile

Argumente:

mm/dd/yy	Gültiges Datum in der Reihenfolge: Monat (mm, 1..12), Tag (dd, 1..31), Jahr (yy, 79...99)
hh:mm	Uhrzeit mit Stunde (hh, 0..23) und Minute (mm, 0..59)
befehlszeile	Beliebige Befehlszeile einschl. Argumenten

Beschreibung:

Der SCHED-Befehl erlaubt die vorprogrammierte Ausführung eines Befehls zu einer bestimmten Zeit. Sobald die MP/M-interne (Software-) Uhr die vorgegebene Zeit erreicht hat, wird die Befehlszeile von der gleichen Konsole aus gestartet. Ist die Konsole nicht frei, so wird der Befehl sofort nach dem Freiwerden ausgeführt. Die vorprogrammierten Schedule-Zeiten gehen beim Ausschalten des Systems verloren.

Anwendung:

Zur Benutzung des SCHED-Programms muß ein residenter Teil (SCHED.SPR) im System eingebaut sein. Beim Aufruf wird das Programm SCHED.PRL geladen. Zur korrekten Arbeitsweise muß natürlich die Software-Uhr gestellt sein (siehe TOD).

Beispiel:

```
0A>SCHED 12/31/84 23:59 PROST↵
```

(Am 31. Dezember 1984 soll um 23:59 Uhr das Programm PROST gestartet werden.)



o CP/M-80
o CP/M-Plus
• MP/M-80
o CP/M-86
• CCP/M
• MP/M-86

Directory (Anzeige des Dateien-Inhaltsverzeichnisses)

Format:

1. SDIR
2. SDIR d:
3. SDIR filespec
4. SDIR [option]
5. SDIR [option] filespec
6. SDIR filespec [option]
7. SDIR [option] filespec, filespec
8. SDIR filespec, filespec [option]

Argumente:

d:	Ein Laufwerk (A...P)
filespec	Dateiname oder Gruppenname unter Benutzung von [?] und [*] zur Einschränkung der angezeigten Dateien. SDIR ohne Argument wird als SDIR *.* interpretiert.
[option]	zusätzliche Informationen zur Steuerung besonderer Funktionen. Folgende Optionen sind implementiert:
[DIR]	Zeigt nur Dateien ohne System-Attribut.
[DRIVE=ALL]	Zeigt die Dateien aller dem System bekannten Laufwerke.
[DRIVE=d]	Zeigt die Dateien des Laufwerks d. Es können auch mehrere Laufwerke angegeben werden (durch Komma getrennt und in runde Klammern gesetzt).
[EXCLUDE]	Zeigt die Dateien, die von der Dateigruppenbezeichnung ausgeschlossen werden.

[FF]	Sendet ein Formfeed-Zeichen zum Drucker, wenn das Drucker-Echo (^P) eingeschaltet ist.
[FULL]	Zeigt die Directory mit allen Informationen (Attribute, Paßwörter, Zeiteinträge) an.
[HELP]	Zeigt eine Liste möglicher SDIR-Befehle.
[LENGTH=n]	Vereinbarung der Zeilenzahl pro Seite (n = 5...65536). Die Kopfzeile der Directory wird auf jeder Seite wiederholt.
[MESSAGE]	Laufwerksnummer und Benutzerbereich, die gerade durchsucht werden, werden angezeigt.
[NONXFCB]	Zeigt nur Dateien ohne erweiterten Directory-Eintrag (XFCB).
[NOSORT]	Die alphabetische Sortierung wird abgeschaltet.
[RO]	Es werden nur die Dateien mit Schreibschutz-Attribut angezeigt.
[RW]	Es werden nur die Dateien ohne Schreibschutz-Attribut angezeigt.
[SIZE]	Es wird jeweils der Dateiname und die Länge in KByte angezeigt.
[SHORT]	Ausgabe der Directory in Kurzform.
[SYS]	Nur Dateien mit System-Attribut werden angezeigt.
[USER=ALL]	Es werden die Dateien aller Benutzerbereiche angezeigt.
[USER=n]	Es werden die Dateien des Benutzer-Bereichs n angezeigt. Es können auch mehrere Benutzernummern angegeben werden (durch Komma getrennt und in runde Klammern gesetzt).
[XFCB]	Zeigt nur Dateien mit erweitertem Directory-Eintrag.

Beschreibung:

SDIR ist ein komfortabler Befehl zur Darstellung der Directory. Zahlreiche Optionen spezifizieren die einzelnen Funktionen.

Anwendung:

Wird nur SDIR eingegeben (Format 1), so wird die Directory des Arbeits-

laufwerks in voller Länge ausgegeben. Mit einer zusätzlichen Laufwerksangabe kann auch ein anderes Laufwerk angewählt werden. Mit Dateinamen oder Dateigruppenbezeichnungen läßt sich die Ausgabe auf diese Dateien einschränken.

Beispiel:

```
0A>SDIR [USER=ALL,DRIVE=(A,B),SYS] *.COM *.PRL↵
```

Dieser Befehl listet alle Dateien vom Typ „.COM“ oder „.PRL“ mit System-Attribut von allen Benutzerbereichen der Laufwerke A und B.

- ○ CP/M-80
- ● CP/M-Plus
- ● MP/M-80
- ○ CP/M-86
- ● CCP/M
- ● MP/M-86



Setzen von Attributen, Paßwörtern usw.

Format:

1. SET [option]
2. SET d:[option]
3. SET filespec [option]

Argumente:

[option]	Eine oder mehrere, durch Kommas getrennte Anweisungen.
d:	Laufwerksangabe (A...P).
filespec	Dateiname oder Dateigruppenbezeichnung, wahlweise mit Laufwerksangabe.

Beschreibung:

SET ist ein Multifunktions-Befehl zum Setzen verschiedener Laufwerks- und Datei-spezifischer Parameter. Die einzelnen Funktionen sollen im folgenden kurz beschrieben werden:

Zuordnung eines Namens zu einer Platte:

SET [d:] [NAME=Labelname.Typ]

Auf der Platte in Laufwerk d wird ein Directory-Label angelegt mit dem Plattennamen Labelname.Typ.

Vereinbarung eines Paßwortes zu einem Label:

SET [d:] [PASSWORD=passwort]

Das Directory-Label der Platte in Laufwerk d wird durch das Geheimwort „passwort“ gegen unerlaubten Zugriff geschützt. Wird die Zeile hin-

ter dem =-Zeichen beendet, so bedeutet das ein Aufheben des Paßwort-Schutzes.

Einschalten des Paßwort-Schutzes für Dateien:

SET [d:] [PROTECT=ON]

Die einzelnen Dateien der Platte d können Paßwort-geschützt werden.

Ausschalten des Paßwort-Schutzes für Dateien:

SET [d:] [PROTECT=OFF]

Es wird keine Paßwort-Überprüfung der Dateien vorgenommen.

Vereinbarung eines Datei-Paßwortes:

SET filespec [PASSWORD=passwort]

Die Datei(en) filespec bekommen das Geheimwort „passwort".

Vereinbarung des Paßwort-Modus:

SET filespec [PROTECT=READ]
SET filespec [PROTECT=WRITE]
SET filespec [PROTECT=DELETE]
SET filespec [PROTECT=NONE]

Es wird vereinbart, für welche Zugriffsart der Paßwort-Schutz gelten soll:

READ	Alle Dateizugriffe sind geschützt.
WRITE	Alle Zugriffe, die Änderungen an Dateien vornehmen (auch Löschen und Umbenennen), sind geschützt.
DELETE	Die Datei ist gegen Löschen und Umbenennen geschützt.
NONE	Der Paßwort-Schutz wird ausgeschaltet.

Vereinbarung eines Arbeits-Paßwortes (default password):

SET [DEFAULT=passwort]

Ein Arbeits-Paßwort kann vereinbart werden. Dieses Paßwort wird bei allen Zugriffen, bei denen ein Geheimwort verlangt wird, mit überprüft.

Vereinbarung von Zeiteinträgen:

SET [CREATE=ON]
SET [ACCESS=ON]
SET [UPDATE=ON]

Die Datei-abhängigen Zeit- und Datumseinträge werden eingeschaltet.

CREATE	Der Zeitpunkt beim Erstellen einer Datei wird eingetragen.
ACCESS	Bei jedem Zugriff wird ein Zeiteintrag gemacht. Das Einschalten der ACCESS-Einträge schaltet automatisch die CREATE-Einträge aus.
UPDATE	Bei jeder Änderung der Datei (Schreibzugriff auf die Directory) wird ein Zeiteintrag gemacht.

CREATE- und ACCESS-Einträge können nicht gleichzeitig eingeschaltet sein.

Setzen von Datei-Attributen:

Option	Bedeutung
ARCHIVE=OFF	Das Archiv-Attribut wird zurückgesetzt.
ARCHIVE=ON	Das Archiv-Attribut wird gesetzt.
DIR	Das System-Attribut wird zurückgesetzt.
Fx=ON/OFF	Das Attribut Fx mit x=1...4 wird gesetzt bzw. zurückgesetzt.
RO	Das Schreibschutz-Attribut (read only) wird gesetzt.
RW	Das Schreibschutz-Attribut wird zurückgesetzt.
SYS	Das System-Attribut wird gesetzt.

Setzen von Laufwerks-Attributen:

SET [d:] [RO]	Das Laufwerk wird in den Schreibschutz-Status gesetzt.
SET [d:] [RW]	Der Schreibschutz-Status für das Laufwerk wird zurückgenommen.

Anwendung:

Das Directory-Label sowie die Datei-abhängigen Eintragungen von Geheimwörtern und Zeiten stehen mit in der Directory. Bei CP/M-Plus müssen die Platten, auf denen diese erweiterten Directory-Einträge (XFCBs) stehen sollen, vorher entsprechend initialisiert werden. Dazu dient das Programm INITDIR.COM.

Beispiele:

```
1A>SET B:[NAME=BRIEFE.TXT]↵
```

(Die Platte in Laufwerk B erhält ein Directory-Label mit dem Namen BRIEFE.TXT.)

```
1A>SET B:[CREATE=ON,UPDATE=ON]↵
```

(Die Dateien auf der Platte in Laufwerk B bekommen in Zukunft Zeiteinträge bei der Erstellung und Änderung.)

```
0A>SET B:*.COM [RO,SYS]↵
```

(Alle Dateien vom Typ „.COM“ auf Laufwerk B erhalten das Schreibschutz- und das System-Attribut.)

o CP/M-80	o CP/M-86
• CP/M-Plus	• CCP/M
• MP/M-80	• MP/M-86



Anzeige Laufwerks- und Datei-spezifischer Parameter

Format:

1. SHOW
2. SHOW d:
3. SHOW option
4. SHOW d:option

Argumente:

d:	Laufwerksbezeichnung (A...P)
option	Zusatz zur Anzeige bestimmter Werte. Bei CP/M-Plus werden die Optionen in eckige Klammern gesetzt. Folgende Optionen sind implementiert:
SPACE	Anzeige des Zugriffs-Modus und des freien Speicherplatzes auf der Platte.
DRIVE	Anzeige der physikalischen Laufwerksparameter.
USERS	Anzeige der benutzten User-Bereiche. Das sind die Bereiche, auf denen Dateien stehen.
LABEL	Anzeige des Directory-Labels.
HELP	Anzeige der möglichen SHOW-Befehle (nicht CP/M-Plus).
DIR	Anzeige der freien Directory-Plätze (nur CP/M-Plus).

Beschreibung:

Der SHOW-Befehl dient zur Anzeige Laufwerks- und Datei-spezifischer Werte. SHOW ohne Parameter gibt den freien Speicherplatz aller dem System bekannten Laufwerke aus. Wird ein Laufwerk angegeben, so beschränkt sich die Ausgabe auf dieses Laufwerk (Format 2).

Bei CP/M-80 und CP/M-86 werden die meisten Funktionen von SHOW von STAT übernommen.

Beispiele:

```
1A>SHOW↵
A:RW, Space:    4k
B:RW, Space: 964k
1A>
```

(Laufwerk A hat noch 4 KByte freien Speicheplatz, Laufwerk B hat noch 964 KByte. Beide Laufwerke sind im Lese-Schreib-Status, also nicht schreibgeschützt.)

```
1A>SHOW [USERS]↵
Active User:  1
Active Files:  0   2  3   4
A: # of files:  95  40  1  26
A: Number of free directory entries: 350
1A>
```

(Ausgabe der benutzten User-Bereiche bei CP/M-Plus. Im Benutzerbereich 0 stehen 95 Dateien, im Benutzerbereich 2 stehen 40 Dateien usw. 350 Plätze für Directory-Einträge sind noch frei.)

o CP/M-80 o CP/M-86
• CP/M-Plus o CCP/M
o MP/M-80 o MP/M-86



Symbolischer 8080-Debugger

Format:

1. SID
2. SID programm,*symboldatei*
3. SID utility

Argumente:

programm — Name einer Datei, i.a. Name eines Programms.

symboldatei — Optionaler Name einer Symbol-Datei, wie sie von verschiedenen Assemblern und Linkern erzeugt wird.

utilty — Name einer Datei vom Typ „.UTL" als Erweiterungs-Routine für SID. Folgende Utilities stehen zur Verfügung:

TRACE — Der TRACE-Zusatz ermöglicht es, den Programmverlauf bis zu 256 Schritte vor einem Haltepunkt (Breakpoint) anzuzeigen (backtrace).

HIST — Mit dem HIST-Zusatz läßt sich eine grafische Darstellung über die Benutzungshäufigkeit von Programmabschnitten ausgeben (Histogramm).

Beschreibung:

SID ist ein symbolischer 8080-Debugger (Testhilfe) mit der Möglichkeit der Arbeit in Assembler-Code und der Benutzung symbolischer Adressen sowie Erweiterung über sogenannte Dienstprogramme.

Anwendung:

Format 1 lädt nur das SID-Programm. SID lädt sich automatisch in den oberen Bereich der TPA. Format 2 bewirkt, daß sofort nach dem Start des

SID das angegebene Programm in die TPA geladen wird. Besteht für dieses Programm eine Symboltabelle als Datei, so kann diese gleich mit geladen werden. Format 3 läßt sich benutzen, um eine SID-Erweiterung zu laden.

SID und die Z80-Version ZSID können von Digital Research auch separat bezogen werden; sie laufen auch auf CP/M-80 und MP/M-80.

- o CP/M-80
- o CP/M-Plus
- • MP/M-80
- o CP/M-86
- o CCP/M
- • MP/M-86



Ausdrucken von Dateien im Hintergrund

Format:

SPOOL dateibez,dateibez,...

SPOOL dateibez [DELETE]

Argumente:

dateibez	Dateiname der auszudruckenden Datei. Es können mehrere Dateien angegeben werden.
[DELETE]	Option, die das Löschen der Datei nach dem Ausdrucken veranlaßt.

Beschreibung:

SPOOL veranlaßt die Ausgabe von Dateien auf den Drucker. Nach dem Start des SPOOL-Programms schaltet es sich in den Hintergrund (detach) und gibt somit die Konsole frei für andere Arbeiten. Der Ausdruck kann jederzeit mit dem STOPSPLR-Befehl abgebrochen werden.

Anwendung:

Bei MP/M-80 ist es möglich, den Spooler als residenten Systemprozeß in das Betriebsystem mit einzubinden. In dieses Fall übergibt der SPOOL-Befehl nur die Argumente in eine Warteschlange des residenten Teils und braucht deshalb nicht im Hintergrund weiterzulaufen.

Beispiel:

```
0A:SPOOL ADRESS.TXT↵
MP/M II V2.0 Spooler
– Enter STOPSPLR to abort the spooler
– Enter ATTACH SPOOL to re-attach console to spooler
***Spooler detached from console***
0A>
```

(Die Datei ADRESS.TXT wird im Hintergrund ausgedruckt. In diesem Fall war kein residenter Systemprozeß vorhanden, so daß SPOOL im Hintergrund weiterarbeitet.)

● CP/M-80	● CP/M-86	
○ CP/M-Plus	● CCP/M	
● MP/M-80	● MP/M-86	



Datei- und Gerätestatus

Format:

1.	STAT	
2.	STAT d:	
3.	STAT d:=RO	(bei CP/M-80: STAT d:=R/O)
4.	STAT filespec	
5.	STAT filespec option	
6.	STAT VAL:	
7.	STAT DEV:	(nur bei CP/M-80)
8.	STAT d:DSK:	
9.	STAT d:USR:	
10.	STAT gen:=dev:,gen:=dev:...	(nur bei CP/M-80)

Argumente:

d:	Laufwerksbezeichnung, d = A...P
d:=RO bzw. d:=R/O	Das Laufwerk d bekommt einen logischen Schreibschutz.
filespec	Dateiname oder Dateigruppenbezeichnung
option	Zusatz zur Steuerung bestimmter Aufgaben. Bei CP/M-80 werden die Attribute von einem $-Zeichen eingeleitet; bei den anderen Versionen stehen sie in eckigen Klammern.
$S [SIZE]	Die virtuelle Dateilänge wird mit ausgelistet. In der Tabelle erscheint ein zusätzliches Size-Feld.
$R/O [RO]	Die in filespec angegebenen Dateien werden in den Schreibschutz-Status (read only) gesetzt.
$R/W [RW]	Der Schreibschutz-Status der in filespec angegebenen Dateien wird aufgehoben (read/write-mode).

$SYS [SYS]	Die in filespec angegebenen Dateien erhalten das System-Attribut.
$DIR [DIR]	Das System-Attribut für die in filespec angegebenen Dateien wird zurückgesetzt.
VAL:	Eine Auflistung der möglichen STAT-Befehle wird gegeben.
DEV:	Die aktuelle Zuordnung der externen Geräteeinheiten (I/O-Byte bei CP/M-80) wird ausgegeben.
d:DSK:	Die wichtigsten Plattenparameter (die für das Betriebssystem von Bedeutung sind) von Laufwerk d werden ausgegeben. Wird d: weggelassen, so werden die Parameter aller z.Z. dem System bekannten Laufwerke aufgelistet.
d:USR:	Die augenblickliche Benutzernummer sowie eine Liste aller auf Laufwerk d benutzten Benutzerbereiche wird ausgegeben.
gen:=dev:	Zuordnung von externen Geräteeinheiten über das I/O-Byte bei CP/M-80. gen: steht für die logische Bezeichnung (CON:, PUN:, RDR:, LST:), dev: für die physikalische (TTY:, CRT: usw.).

Beschreibung:

STAT ist ein Multifunktions-Programm zur Abfrage und Änderung verschiedener Datei-, Laufwerks- und Gerätezustände. Format 1 und 2 werden verwendet, um den freien Speicherplatz auf der Platte festzustellen. Mit Format 3 läßt sich ein Laufwerk in einen vorübergehenden logischen Schreibschutz-Status bringen. Format 4 erzeugt eine Liste der in filespec bezeichneten Dateien mit Angabe von Dateilänge und Attributen. Format 5 läßt sich, bei Verwendung der SIZE-Option, zur Erweiterung der Ausgabe von Format 4 einsetzen, oder, bei Verwendung der anderen Optionen, zur Änderung von Datei-Attributen.

Anwendung:

Folgende Besonderheiten müssen bei der Anwendung beachtet werden:

1. Ist ein Laufwerk durch Diskettenwechsel in den logischen Schreibschutz-Status gesetzt worden, stimmt die Angabe des freien Speicherplatzes mit STAT nicht mehr.
2. Die Zuordnung externer Geräteeinheiten unter CP/M-80 funktioniert nur, soweit sie im BIOS implementiert ist.

Eine Beschreibung des STAT-Befehls finden Sie auch in Kapitel 2.

Bei neueren Systemen übernehmen die Befehle SHOW und SET die Aufgaben von STAT.

Beispiele:

```
A>STAT↵
A:R/W,SPACE:144K
A>
```

(Das Laufwerk A ist nicht schreibgeschützt und hat 144 KByte freien Speicherplatz.)

```
A>STAT *.COM↵

Recs  Bytes  Ext  Acc
  48     6K    1  R/O A:ED.COM
  55    12K    1  R/O (A:PIP.COM)
 140    18K    2  R/W A:BASIC.COM
Bytes Remaining on A: 116K

A>
```

(Auf dem Laufwerk A befinden sich 3 Dateien vom Typ „.COM“. Die Spalte „Recs“ gibt die Dateilänge in 128-Byte-Records an, die Spalte „Bytes“ den von der Datei blockierten Speicherplatz in KByte, die Spalte „Ext“ die Anzahl der Directory-Einträge und die Spalte „Acc“ den Zugriffs-Modus. Der dann folgende Dateiname wird in Klammern gesetzt, wenn das System-Attribut gesetzt ist.)



- ○ CP/M-80
- ○ CP/M-Plus
- ● MP/M-80
- ○ CP/M-86
- ○ CCP/M
- ● MP/M-86

Abbruch der SPOOL-Operation

Format:

STOPSPLR

Beschreibung:

Der Befehl STOPSPLR unterbricht die laufende SPOOL-Operation und löscht die SPOOL-Warteschlange (vergl. SPOOL-Befehl).

Beispiel:

```
0A>STOPSPLR↵
```

- CP/M-80
- CP/M-Plus
- MP/M-80

- CP/M-86
- CCP/M
- MP/M-86



Einleitung eines automatischen Befehlsablaufs (Batch)

Format:

SUBMIT dateiname v1 v2 v3 ...

Argumente:

dateiname	Name der Textdatei mit den zu verarbeitenden Befehlszeilen. Wird kein Dateityp angegeben, so nimmt SUBMIT den Dateityp „.SUB“ an.
v1 v2 v3 ...	Wahlweise angebbare Variablenwerte. In der Submit-Textdatei sind die entsprechenden Variablen gekennzeichnet mit \$1, \$2, \$3 usw.

Beschreibung:

Der SUBMIT-Befehl dient zur Einleitung eines automatischen Befehlsablaufs (Batch-Processing). Die Eingangsdatei besteht aus einer Folge von Befehlszeilen. SUBMIT bildet daraus eine Datei mit der Bezeichnung „$$$.SUB“, in der die ggf. benutzten Variablen substituiert sind. Sie wird dann vom Befehlsprozessor eingelesen und genauso abgearbeitet wie eine Eingabe von der Konsole. Nach vollständiger Abarbeitung oder Abbruch wird die Datei gelöscht. Zeileneingaben für die gestarteten Programme sind auch möglich, jedoch muß bei CP/M-80 der Befehl XSUB vorher in der Befehlsdatei stehen bzw. bei CP/M-Plus vor jeder dieser Eingaben das Zeichen < gesetzt sein.

Anwendung:

Der gewünschte Ablauf wird mit einem Editor in eine Textdatei geschrieben, die dann bei SUBMIT als Argument angegeben wird. Bei CP/M-80 wird die $$$.SUB-Datei auf dem Arbeitslaufwerk angelegt. Sie wird

jedoch nur gestartet, wenn sie auf Laufwerk A steht. Bei der Erstellung der Textdatei muß darauf geachtet werden, daß keine Leerzeilen im Text stehen (auch nicht am Dateiende) und daß die Gesamtlänge 128 Zeilen nicht überschreitet.

Beispiel:

Die Befehlsdatei TEST.SUB enthält folgende Textzeilen:

```
DIR $1.*
PIP $2:=$1.BAK
ERA $1.BAK
```

Die Ausführung der einzelnen Befehlszeilen kann nun vom Benutzer mit dem folgenden Submit-Befehl gestartet werden:

```
A>SUBMIT TEST PROG B↵
```

Die Befehlszeilen werden in der Datei $$$.SUB mit den Werten der Variablen wie folgt transformiert:

```
DIR PROG.*
PIP B:=PROG.BAK
ERA PROG.BAK
```

Bei der Abarbeitung der $$$.SUB-Datei werden diese Befehlszeilen mit auf die Konsole ausgegeben.

● CP/M-80
o CP/M-Plus
o MP/M-80
o CP/M-86
o CCP/M
o MP/M-86



Lesen und Schreiben der Systemspuren

Format:

1. SYSGEN
2. SYSGEN dateiname

Argument:

dateiname — Name einer Datei, die das System enthält. Diese Datei muß mit dem SAVE-Befehl erstellt sein, da SYSGEN sie erst ab dem 17. Record zur Adresse 900H lädt.

Beschreibung:

Das SYSGEN-Programm liest und beschreibt die Systemspuren von 8-Zoll-Single-Density-Disketten. Wird beim Aufruf ein Dateiname angegeben, so wird das System anstatt von den Systemspuren von dieser Datei in die TPA geladen. SYSGEN kann auch durch Überspringen der Leseoperation ein bereits in der TPA stehendes System auf die Systemspuren schreiben.

Anwendung:

SYSGEN arbeitet im Dialogbetrieb. Die Bezeichnung des Quellen- und des Ziellaufwerks kann vom Benutzer auf Anforderung hin eingegeben werden. Es ist möglich, nacheinander mehrfach das Ziellaufwerk anzugeben, um mehrere Kopien zu erstellen.

Zum Kopieren der Systemspuren von anderen als 8-Zoll-Single-Density-Disketten liefern vielfach die Rechnerhersteller entsprechende Programme mit. Bei CP/M-Plus ist ein Programm COPYSYS.COM im Lieferumfang enthalten.

TOD

- o CP/M-80
- o CP/M-Plus
- • MP/M-80
- o CP/M-86
- • CCP/M
- • MP/M-86

Lesen und Setzen von Uhrzeit und Datum

Format:

1. TOD *P*

2. TOD *mm/dd/yy hh:mm:ss*

Argumente:

P — Wahlweise angebbares Argument, das die fortwährende Anzeige von Uhrzeit und Datum veranlaßt.

mm/dd/yy — Argument zum Setzen des Datums in der Reihenfolge: Monat (01...12), Tag (01...31) und Jahr (79...99).

hh:mm:ss — Argument zum Setzen der Uhrzeit in der Reihenfolge: Stunde (00...23), Minute (00...59) und Sekunde (00...59).

Beschreibung:

TOD wird zum Lesen und Stellen der systemeigenen Softwareuhr verwendet. Diese Uhr wird z.B. gebraucht, um die Directory-Einträge mit Zeitinformationen zu versehen (time and date stamps).

Anwendung:

Wird nur TOD eingegeben, so werden Wochentag, Datum und Uhrzeit auf der Konsole ausgegeben. Mit dem Zusatz P läßt sich auf dem Bildschirm eine laufende Digitaluhr darstellen (durch Eingabe irgendeines Zeichens abzubrechen). Zum Stellen der Uhr wird die gewünschte Zeit als Argument mit eingegeben. Um ein Sekunden-genaues Setzen zu gewährleisten, wartet TOD bis zur Eingabe eines weiteren Zeichens.

Bei CP/M-Plus wird TOD durch das Programm DATE.COM ersetzt.

Beispiel:

```
0A>TOD↵
Sun 01/29/84 13:42:45
0A>TOD 01/29/84 13:46:00↵
Strike key to set time
Sun 01/29/84 13:46:00
0A>
```

TYPE

- CP/M-80
- CP/M-Plus
- MP/M-80
- CP/M-86
- CCP/M
- MP/M-86

Ausgabe einer Textdatei auf die Konsole

Format:

1. TYPE dateibez
2. TYPE dateigruppe [option] (nur bei CP/M-Plus und MP/M)

Argumente:

dateibez	Name einer ASCII-Textdatei
dateigruppe	Name einer Datei oder Dateigruppe mit ASCII-Textinhalt.
[option]	Optionaler Parameter zur Steuerung der seitenweisen Ausgabe. Folgende Optionen sind implementiert:
[PAGE]	Ausgabe stoppt nach jeweils 24 Zeilen.
[NO PAGE]	Ausgabe stoppt nicht (nur CP/M-Plus).
[Pn]	Ausgabe stoppt nach jeweils n Zeilen (nur MP/M).

Beschreibung:

Der TYPE-Befehl gibt den Inhalt der angegebenen ASCII-Datei auf die Konsole aus. Tabulations-Zeichen werden dabei automatisch in Leerzeichen umgewandelt, wobei für jede 8. Spalte ein Tabulator-Haltepunkt angenommen wird.

Anwendung:

TYPE ist bei CP/M ein residenter Befehl, während er bei den anderen Systemen als Befehls-Datei auf der Platte sein muß. Die Ausgabe von TYPE kann mit ^P (Drucker-Echo) gleichzeitig auf den Drucker gelegt werden. Mit ^S und ^Q läßt sich die Ausgabe stoppen und wieder starten. ^C (und beim Standard-CP/M jede andere Eingabe) bricht den TYPE-Befehl ab.

Beispiel:

```
A>TYPE REST.SUB↵
1A>TYPE *.TXT↵
```

(Im ersten Fall wird die Datei REST.SUB ausgegeben; im zweiten Fall z.B. nacheinander die Dateien BRIEF1.TXT, RECHNUNG.TXT, BELEG.TXT.)



- CP/M-80
- CP/M-Plus
- MP/M-80
- CP/M-86
- CCP/M
- MP/M-86

Umschalten des aktuellen Benutzerbereichs

Format:

1. USER (nicht bei CP/M-80)
2. USER n

Argument:

n Nummer des Benutzerbereichs (0...15).

Beschreibung:

Der USER-Befehl dient zur Anwahl eines Benutzerbereichs. Der Standardbereich ist 0 und wird bei CP/M nach dem Kaltstart automatisch angewählt. Der augenblicklich aktive Benutzerbereich wird bei MP/M und CCP/M bei der System-Meldung mit angezeigt; bei CP/M-Plus nur, wenn er nicht 0 ist.

Anwendung:

Der USER-Befehl ist bei CP/M resident; bei den anderen Systemen muß er als Befehlsdatei auf der Platte stehen. Wichtig ist es, daß diese Datei dann entweder auf jedem Benutzerbereich steht oder aber im User-Bereich 0 als System-Datei, da sonst aus dem neuen Bereich nicht wieder zurückgeschaltet werden kann.

Beispiel:

```
A>USER 3↵
3A>
```

(Benutzerbereichs-Umschaltung bei CP/M-Plus von 0 auf 3.)

- ● CP/M-80
- o CP/M-Plus
- o MP/M-80
- o CP/M-86
- o CCP/M
- o MP/M-86



Erweiterung der SUBMIT-Funktion

Format:

```
XSUB
```

Beschreibung:

Der SUBMIT-Befehl bei CP/M-80 dient nur zur Abarbeitung von CP/M-Befehlszeilen. XSUB ist ein Programm, das auch die von Programmen benötigten Eingabezeilen aus der $$$.SUB-Datei entnimmt. Es muß dazu als erster Befehl innerhalb der SUBMIT-Datei stehen.

Anwendung:

Bei der Verwendung von XSUB muß darauf geachtet werden, daß keine Einzelzeichen verarbeitet werden (nur Zeileneingaben) und daß der freie TPA-Raum verkleinert wird.

Beispiel:

Die Datei NEU.SUB enthält folgenden Inhalt:

```
XSUB
DDT
I$1.HEX
R
G0
SAVE 1 $2.COM
```

Bei Aufruf von

```
A>SUBMIT NEU DIES DAS↵
```

wird DDT geladen, die Datei DIES.HEX in den Speicher gebracht (und in Binärwerte gewandelt), dann ein Warmstart durchgeführt und der TPA-Inhalt als DAS.COM auf die Platte gesichert.

Kapitel 7

Praktische Hinweise

EINFÜHRUNG

Nachdem nun die wesentlichen Funktionen und Möglichkeiten von CP/M erläutert worden sind, sollen für die effektive Nutzung des CP/M-Gesamtsystems noch einige wichtige Hinweise und Erläuterungen gegeben werden. In diesem Kapitel wird beschrieben, wie die bei der Anwendung eines CP/M-Systems häufig vorkommenden Probleme vermieden bzw. leicht gelöst werden können.

Beim Einsatz eines Computersystems muß dem Schutz vor falscher Bedienung besondere Bedeutung beigemessen werden. Sehr einfache, leicht entstehende Fehlbedienungen durch den Benutzer haben häufig schwerwiegende Fehler zur Folge. Eine hohe Benutzerdisziplin ist deshalb unbedingt einzuhalten.

BENUTZERDISZIPLIN

Immer wenn eine Kopie eines neuen Programms oder einer ganzen Diskette zum erstenmal erstellt wird, muß sehr genau darauf geachtet werden, daß die Originalprogramme nicht versehentlich überschrieben oder gelöscht werden. Wenn eine Diskette in ein Laufwerk eingeschoben wird, muß dies im Fall von CP/M 2.2 durch die Eingabe von CTRL-C dem System mitgeteilt werden (logged in). Von Zeit zu Zeit sollte ein STAT- oder DIR-Befehl ausgeführt werden, um so die Directory zu überprüfen und den noch freien Platz auf der Diskette zu ermitteln.

Bei Abschluß eines langen Editiervorgangs sollte von der generierten oder veränderten Datei oder der gesamten Diskette unbedingt eine Kopie erstellt werden.

Für den Computerraum empfiehlt sich eine räumliche und organisatorische Ausstattung, die ebenfalls eine hohe Benutzerdisziplin gewährleistet (vgl. Listc typischcr Raumausstattungen im Anhang). Alle Benutzerdokumentationen, leere Disketten und Anwenderprogramme sollten jederzeit verfügbar sein. Genauso sollten wichtige Hinweise zur Systemnutzung klar und eindeutig bereit liegen (Angaben zum Systemstart – Hinweise zur Fehlerbehandlung).

BEHANDLUNG VON DISKETTEN

Bei der Nutzung von Disketten müssen die physikalischen Eigenschaften der Diskette sorgfältig berücksichtigt werden. Insbesondere muß auf folgendes geachtet werden:

- Keine magnetischen Objekte dürfen in die Nähe von Disketten oder Disketten in die Nähe von Magnetfeldern gebracht werden. Hierzu gehören Transformatoren, Telefonapparate, Schraubenzieher (die meisten haben ein geringes magnetisches Feld) oder andere metallische und magnetische Objekte.
- Disketten dürfen nicht geknickt, verdreht oder sonst mechanisch beschädigt werden.
- Disketten dürfen nicht Wärmequellen oder dem direkten Sonnenlicht ausgesetzt werden.
- Disketten sollten nur mit einem Filzschreiber oder einer weichen Schreibfeder beschriftet werden. Beschriftungen mit einem harten Bleistift oder einem Kugelschreiber können die Schutzhülle oder die Diskette direkt zerstören.
- Disketten sollten hinsichtlich Datum und Inhalt vollständig gekennzeichnet sein. Sehr häufig wird zusätzlich von Benutzern nach der Bearbeitung einer Diskette eine Directory ausgedruckt und diese mit der Diskette in die Schutzhülle geschoben. Auf diese Weise kann der Disketteninhalt leicht ermittelt werden.
- Von wichtigen Disketten müssen Kopien an anderen Orten als das Original aufbewahrt werden, um so das Risiko des vollständigen Informationsverlustes so gering wie möglich zu halten.
- Disketten dürfen beim Einschalten oder Abschalten der Netzversorgung nicht vollständig in die Laufwerke eingeschoben sein.
- Von allen wichtigen Dateien und Disketten sollten Kopien erstellt werden.

Disketten müssen sehr sorgfältig vor Verschmutzung geschützt werden. Dies gilt besonders in trockenen Räumen. Die auf der Diskette entstehende Reibungselektrizität zieht Abrieb- und andere Staubpartikel statisch an, wodurch sehr leicht Beschädigungen entstehen können. Wichtige Regeln sind:

- Disketten sollten niemals in die Nähe einer Staubquelle oder eines potentiellen Stauborts gelegt werden. Solche Staubquellen und Stauborte sind z.B. Lüfter, Aschenbecher, Taschen, Schubladen, Teppiche, Gardinen usw.
- Die Rechner-Umgebung kann durch Verwendung spezieller Reinigungsmittel, die nach dem Vakuumprinzip arbeiten, weitgehend staubfrei gehalten werden. Auch können in die Raumheizungs- und Kühlsy-

steme spezielle Filter eingesetzt oder elektronische Luftreiniger verwendet werden.

- Die Bildung elektrischer Felder im Rechnerraum sollte unbedingt vermieden werden. Hierzu kann beispielsweise ein Antistatikspray auf die Teppich-Oberfläche gesprüht werden. Wesentlich wirkungsvoller ist jedoch ein Luftbefeuchter.
- Genauso wichtig ist natürlich die möglichst ständige Aufbewahrung der Disketten in den dazugehörigen Schutzhüllen.

Die Schreib/Lese-Köpfe der Laufwerke verschmutzen durch den Abrieb von den Floppies. Für die Reinigung werden Reinigungssets angeboten.

AUSGABE

Der Drucker

Qualitativ hochwertige Drucker arbeiten im allgemeinen monatelang oder jahrelang sehr zuverlässig und ohne Ausfall. Voraussetzung ist jedoch eine sorgfältige Behandlung. Alle mechanischen, vom Benutzer durchzuführenden Einstellungen müssen dabei genauestens ausgeführt werden. Vom Benutzer muß deshalb verlangt werden, daß er die wichtigen Bedienungsvorgänge kennt und korrekt erledigt. Hierzu gehören bestimmte Einstellungen zum Papiervorschub, zur Papierlänge, Papierdicke, Farbbandjustierung usw.

Als Beispiel kann hier die Einstellung der Papierdicke gelten. Wird ein sehr dünnes Papier bedruckt, so muß z.B. bei einem IBM-Drucker der Hebel in der Stellung für dünnes Papier (Stellung A) stehen. Ist dieser Hebel versehentlich auf die Stellung D geschaltet (für dickes Papier), so arbeitet der Drucker unzuverlässig und fehlerhaft. Die bei dieser Einstellung erzielten Fehldrucke deuten zunächst auf ein Software- oder Interfaceproblem hin, obwohl sie sehr einfach durch Umstellung des Schalters in Stellung A beseitigt werden können. Andere fehlerhafte Grundeinstellungen können zu ähnlichen Fehlern führen.

Ausgabe auf einen Drucker

Das Ausdrucken von Listings nimmt meist eine lange Zeit in Anspruch, da der Drucker im allgemeinen der wesentliche Engpaß des Gesamtsystems ist. Aus diesem Grunde benutzen viele Anwender diese Zeit als Gelegenheit zu einer Pause und verlassen den Rechnerraum. Hierbei muß jedoch gewährleistet sein, daß eine häufige Überprüfung des korrekten Druckbetriebs erfolgt.

Dies ist besonders wichtig zu Beginn eines Ausdrucks, wenn das Papier in den Einzugschacht hineingezogen wird (Vorsicht bei Selbstklebeetiket-

ten). Werden Farbbänder auf Karbonbasis (einmal verwendbar) benutzt, so bricht das Druckbild bei Erreichen des Bandendes abrupt ab (bei einigen Druckern wird ein Weiterdrucken verhindert). Ein Teil des Ausdrucks bleibt dann ohne Druckfarbe und somit völlig leer. Wenn keine höhere Druckqualität erforderlich ist, empfiehlt sich deshalb die Verwendung von Endlosfarbbändern (auch kostengünstiger).

Wenn Etiketten gedruckt werden, sollte der Bediener (Operator) während der gesamten Druckzeit anwesend sein. Fehlfunktionen können zu einem Versetzen aller folgenden Drucke auf den Etiketten führen. In vielen Fällen kann der Druckbetrieb kurz unterbrochen, die Anfangsstellung neu justiert und danach der Druckbetrieb wieder aufgenommen werden. Wesentlich schwerwiegender sind mechanische Fehler bei der Etikettenführung. In diesem Fall kann eine Zerstörung des Druckers nur durch sofortiges Abschalten verhindert werden.

Wenn ein fehlerhafter Druck erkannt wird, muß der Ausdruck im allgemeinen neu gestartet werden. Wenn die Datei sehr lang ist, kann aus der Druckdatei der noch nicht gedruckte Dateianteil mit dem PIP- oder ED-Befehl selektiert werden (vgl. Kapitel 3 und 4) und so viel Zeit gewonnen werden. Eine andere Möglichkeit ist das erneute Starten des Listings über das Bildschirmterminal und das gezielte Zuschalten des Druckers mit CTRL-P bei Erkennen des noch nicht gedruckten Dateiinhalts.

Diese Möglichkeit ergibt sich mit dem TYPE-Befehl. Spezielle Druckprogramme erlauben dieses Parallelschalten des Druckers durch die Eingabe von CTRL-P im allgemeinen leider nicht.

Wenn der Drucker trotz Zuschalten des Systems nicht oder nicht korrekt arbeitet, sollten zunächst alle Grundeinstellungen des Druckers überprüft werden (z.B. könnte der Drucker auf die Betriebsart „local" gesetzt sein).

Wenn hierbei keine Fehleinstellungen feststellbar sind, dann ist zu prüfen, ob die verwendete CP/M-Version den zugeschalteten Drucker bedienen kann (entsprechender Druckertreiber muß verfügbar und zugeordnet sein).

Wenn mehrere lange Listings ausgedruckt werden sollen, kann der Operator einen entsprechenden Gruppendruckbefehl eingeben und sich so anderen Aufgaben zuwenden (z.B. durch Zuordnen der PRN-Einheit).

Für das schnelle Ausdrucken von Listings kann der PIP (im Gegensatz zum TYPE) benutzt werden, wobei als Listeinheit entweder das Bildschirmterminal mit CON: oder der Drucker mit LST: zugeordnet wird.

Während des Druckens wird bei den meisten Druckern Papierstaub abgerieben. Die Drucker sollten deshalb gelegentlich mit Hilfe eines Staubsaugers vorsichtig gereinigt werden.

DATEIEN

Dateiüberlauf (File Overflow)

Wenn die Größe einer Datei die Kapazität einer Diskette überschreitet, so muß diese auf eine zweite oder mehrere Disketten abgelegt werden. Im allgemeinen wird dies nicht erst dann organisiert, wenn die erste Diskette überläuft. Es sollte versucht werden, die erste Diskette nach einigen wichtigen Kriterien zu klassifizieren und identifizieren. Wenn z.B. diese Datei eine Liste von Namen und Adressen beinhalten soll, so kann diese z.B. alphabetisch oder nach der Postleitzahl geordnet werden. Wird die alphabetische Sortierung nach Namen gewählt, so können z.B. auf der ersten Diskette alle Daten der Personen mit den Anfangsbuchstaben A bis L und auf der zweiten Diskette alle Daten der Personen mit den Anfangsbuchstaben M bis Z abgelegt werden. In solchen Anwendungen sollte jede Diskette etwa nur halb beschrieben werden, weil nachträglich durchgeführte Sortierungen durch die Sortierdateien den Umfang der Dateien auf der Diskette stark steigern können. Dieser Engpaß bei nachträglichen beliebigen Sortierungen kann vielfach nur bei Festkopfplatten mit hoher Kapazität vermieden werden.

Eine andere Möglichkeit, die Entstehung zu großer Dateien zu verhindern, ist die Aufteilung in Unterdateien. So können beispielsweise die Namen und Adressen von Kunden und Lieferanten auf separaten Disketten untergebracht werden.

An dieser Stelle ist noch auf ein anderes Problem hinzuweisen. Angenommen, es besteht eine große Datei, z.B. 170 kByte, die sortiert werden soll. Das Sortierprogramm befindet sich auf der Diskette in Laufwerk A, die ebenfalls mit 120 kByte Speicherplatz belegt ist. Für das Sortieren werden dann von vielen Sortierprogrammen mindestens weitere 170 kByte freier Speicherraum auf der Diskette verlangt, so daß bei Standarddisketten der Sortiervorgang nicht ohne weiteres durchgeführt werden kann.

Hierzu bietet sich eine einfache Lösung in der Weise an, daß eine neue Systemdiskette mit der einzigen Datei des Sortierprogramms erstellt wird. In diesem Fall ist auf der Systemdiskette genügend Raum für die Sortierdatei. Läuft das Sortierprogramm nicht korrekt ab, so besteht nur noch die Möglichkeit, die Quelldatei in zwei Dateien aufzuteilen, diese zu sortieren und danach mit dem PIP wieder aneinanderzufügen.

Verketten von Dateien (Merging Files)

Zwei oder mehrere Dateien lassen sich mit dem PIP-Programm beliebig zusammenfügen.

Zerstörte Dateien

Bei einigen Bedienungs- oder Systemfehlern können Dateien zerstört werden. Dies geschieht z.B., wenn innerhalb eines Programmablaufs

vom Benutzer nicht erlaubte Steuerzeichen eingegeben oder wenn vom Programm selbst fehlerhafte Codes erzeugt werden. Im Ergebnis macht sich dies so bemerkbar, daß die Dateien bzw. Programme nicht mehr geladen bzw. gestartet werden können. In den Fällen, in denen die Datei- oder Programmstruktur bekannt ist und beherrscht wird oder der Dateiinhalt vorwiegend aus Text besteht, ist eine Berichtigung leicht möglich. Sind die Dateien nur kurz, so empfiehlt sich jedoch eine erneute Eingabe.

Wenn eine Datei längere Zeit ohne Fehler und Beanstandungen bearbeitet worden ist und plötzlich ein Fehler erkannt wird, so deutet das meist auf einen Bedienungsfehler hin. Genauso häufig sind Fehler auf der System- oder Programmdiskette. Ist die Systemdiskette zerstört, so sind die auf ihr enthaltenen Informationen fehlerhaft, was zu einem fehlerhaften Systemverhalten führt, obwohl die Anwendungsprogramme meist noch fehlerfrei arbeiten. Fehlerhafte Systemdisketten können häufig nicht definierte Systemoperationen generieren, die dann in der Folge wichtige Dateien und Programme zerstören. Aus diesem Grunde sollten bei Erkennen solcher Systemfehler unbedingt neue Systemdisketten bzw. Programmdisketten erstellt werden. Auf diese Weise können Folgefehler vermieden und die effektiven Fehlerursachen leicht erkannt werden.

WICHTIGE DIENSTPROGRAMME

Editor

In Kapitel 4 wird erläutert, wie ein Editor als Werkzeug zur Dateierstellung und -bearbeitung benutzt werden kann. Die dort beschriebenen Funktionen werden vom ED-Programm voll abgedeckt. Darüber hinausgehende, kommerziell einsetzbare Editoren sind für CP/M verfügbar. Sie zeichnen sich besonders durch ihre vielfältigen Befehle aus. Viele wichtige Hinweise zu diesem Gebiet sind in dem Buch von H. Glatzer, *Einführung in die Textverarbeitung,* SYBEX-Verlag, enthalten.

Spur-zu-Spur-Kopierer

Im allgemeinen werden vom Hersteller des Rechners bzw. des Disk-Controllers mehrere Dienstprogramme zur Verfügung gestellt. Hierzu gehört häufig auch ein Kopierprogramm, mit dem der Inhalt einer Diskette oder Platte Spur für Spur auf eine andere Diskette bzw. Platte kopiert werden kann. Hierdurch ergeben sich wesentlich kürzere Kopierzeiten. Ein anderes Dienstprogramm ist z.B. ein Disketten- oder Platteneditor, mit dem einzelne Bereiche auf diesen Datenträgern direkt geändert werden können.

Löschen einer Diskette

Eine Diskette kann aus zwei Gründen gelöscht werden. Einmal, weil

deren Inhalt nicht mehr benötigt wird, und zum anderen, weil die Directory undefiniert geändert wurde und unbrauchbar geworden ist (die Dateiinhalte sind zwar noch in der ursprünglichen Form vorhanden, die physikalischen Speicheradressen jedoch nicht bekannt).

Eine noch benutzbare Diskette kann mit dem Befehl ERA *.* gelöscht werden. Ist die Directory der Diskette jedoch zerstört, so kann sie nur mit einem Formatierprogramm gelöscht werden. Ein solches Programm wird häufig mit dem Namen FORMAT bezeichnet, vom Hersteller des Computers oder des Disk-Controllers geliefert und bezieht sich direkt nur auf den verwendeten Rechner- oder Laufwerktyp. Mit einem solchen Programm können Dateien sehr schnell gelöscht werden.

Befehlsfolgen

Wenn eine gleiche Folge von Befehlen häufig bearbeitet wird, so sollte hierfür eine SUBMIT-Datei generiert werden. Diese Datei besteht aus einer Folge einzelner Befehle und muß mit einem Editor erstellt werden. Wird der Befehl SUBMIT mit dem Namen der Befehlsdatei eingegeben, so wird die Befehlsfolge automatisch ausgeführt.

Wird zum Beispiel ein bestimmtes Programm sehr häufig ausgeführt und benötigt dieses Programm vorher die Eingabe mehrerer Standardparameter, so kann die Befehlsfolge durch die Erstellung einer Befehlsdatei START.SUB automatisiert werden. Der Benutzer braucht dann für die Ausführung des Programms nur noch den Befehl

```
SUBMIT START↵
```

einzugeben.

Stop

Beim Auftreten irgendwelcher System- oder Peripheriefehler muß die Möglichkeit des sofortigen Programmabbruchs bestehen. Hierbei braucht nicht sofort das Netz abgeschaltet zu werden. Vielfach genügt die Eingabe von CTRL-C oder, wenn dies keinen Erfolg zeigt, das Drücken der Reset-Taste. Zu beachten ist jedoch hierbei, daß wichtige Informationen im Arbeitsspeicher verloren gehen. Dateien auf Disketten werden nicht zerstört. Zur Unterbrechung des Druckbetriebs genügt dessen alleinige Abschaltung.

Weiterer Hinweis

Das CP/M-System selbst belegt, wie bereits erläutert, keinen zusätzlichen Speicherplatz auf der Diskette. Es wird auf den hierfür reservierten ersten beiden Spuren gespeichert. Aus diesem Grunde schreiben viele Benutzer das CP/M-System grundsätzlich auf alle Disketten. Das hat den Vorteil,

daß von all diesen Disketten das System geladen und gestartet werden kann. Außerdem wird das Risiko der Zerstörung einzelner Dateien bei der Anwendung eines fehlerhaften PIP-Befehls gemindert. Weil das PIP-Programm in seiner Bedeutung gleich hinter der CP/M einzuordnen ist, wird auch dieses Programm häufig auf alle Disketten kopiert.

HELFER IN DER NOT

Durch Bedienungsfehler, Software- oder Hardwarefehler werden manchmal auch Informationen zerstört. Sehr oft ist die Restauration der Dateien mit einem sehr hohen Aufwand verbunden. Andererseits besteht auch manchmal der Verdacht, daß ein Hardwarefehler aufgetreten ist oder eine Diskette selbst einen (z.B. mechanischen) Fehler aufweist. Hinzu kommt, daß solche Situationen oft gerade dann auftreten, wenn die Zeit sehr knapp ist. Ein Fachmann steht meist nicht sofort zur Verfügung oder ist vergleichsweise teuer. In einer solchen Situation können u.U. die im folgenden kurz beschriebenen Programme wertvolle Hilfe leisten.

Disk Doctor

Werden durch CP/M Schreib- oder Lesefehler von einer Diskette gemeldet, so möchten Sie möglichst viele Informationen auf einen anderen Datenträger retten. Dies bezieht sich nicht nur auf die noch einwandfrei lesbaren Dateien, sondern auch auf die teilweise zerstörten Dateien. Ein Sonderfall, der auch durch den Disk Doctor behandelt wird, ist das Aufheben eines ERA-Befehls (Datei-Löschbefehl) unmittelbar nach dessen Ausführung. Es werden 5 Bearbeitungsstufen (Ward) angeboten:

Ward A Alle zur Zeit nicht benutzten Blöcke werden auf Fehler untersucht. Fehlerhafte Blöcke werden einer speziellen Datei mit dem Namen MORGUE.NOX zugeordnet (ähnliches Verfahren wie beim Programm ZAP).

Ward B Diese Funktion gestattet das Kopieren von Blöcken oder Sektoren von der fehlerhaften Diskette auf eine andere Floppy. Der Benutzer wählt die betreffenden physikalischen Adressen beispielsweise mit Hilfe der Einträge in der Directory.

Ward C Mit Hilfe dieser Bearbeitungsstufe werden alle Sektoren kopiert. Dabei werden fehlerhafte Sektoren durch 0E5H (dezimal 229) oder durch 20H (Zeichen „Blank“) ersetzt, je nachdem ob der Sektor in der Directory oder in dem Datenbereich liegt.

Ward D Mit Hilfe von Ward D können gelöschte Files wieder aktiviert werden. Dies ist jedoch nur sinnvoll, falls zwischen dem Löschen (ERA-Befehl) und dem Aufruf von Ward D keine Schreiboperationen auf der Diskette durchgeführt wurden.

Ward E Alle Directory-Einträge (also auch diejenigen, die einer bereits gelöschten Datei zugeordnet waren) werden angezeigt.

Die Bedienung des Programms Disk Doctor wird durch Erklärungen (Help-Funktion) und eine einfache Menütechnik unterstützt. Mit Ausnahme von Ward B kann dieses Programm auch ohne fundierte Kenntnisse des internen Aufbaus von CP/M benutzt werden.

Spezifikationen:

Programmname:	Disk Doctor
Aufgabe:	Bearbeitung teilweise zertörter Dateien
Hersteller:	SuperSoft Inc.
Betriebssytem:	CP/M 2.2
Prozessor:	8080, 8085, Z80
Min. Speicher:	k.A.
Anpassung:	Anpassung an viele Floppy- und Winchester-Formate durch den Benutzer
Bedienerführung:	Help-Funktion und einfache Menüs

Diagnostic II

Insbesondere bei komplizierten Software-Fehlern taucht oft der Verdacht eines Hardwarefehlers auf. In einem solchen Fall suchen Sie eine Möglichkeit, Ihre Hardware zu testen. Nicht zuletzt ist es auch im Fall eines echten Hardwarefehlers sehr angenehm, wenn der Fehler schon grob eingekreist werden kann, bevor der technische Service angesprochen wird. Das Programmpaket Diagnostic II bietet separate Testprogramme für

Speicher,
Prozessor (CPU),
Laufwerk bzw. Disketten,
Bildschirm-Terminal und
Drucker.

Mit Ausnahme von Terminal und Drucker laufen alle Tests automatisch ab (bei den Peripheriegeräten ist eine visuelle Kontrolle erforderlich).

Nach dem Aufruf eines Programms wie z.B. durch

```
A>MTEST↵
```

werden die erforderlichen Parameter vom System abgefragt. Im Fall von Diagnostic II können die Tests auch vollautomatisch mit Hilfe von SUBMIT-Files ablaufen.

Die (englische) Dokumentation ist ausführlich und verständlich.

Spezifikationen:

Programmname:	Diagnostic II
Aufgabe:	Test von Speicher, Prozessor, Plattenlaufwerken, Terminal und Drucker
Hersteller:	SuperSoft Inc.
Betriebssytem:	CP/M 2.2
Prozessor:	8080, 8085, Z80
Min. Speicher:	k.A.
Anpassung:	nicht erforderlich
Bedienerführung:	Frage/Antwort nach dem Start

Reclaim

Das Programm Reclaim gestattet das Prüfen von Disketten und Festplatten. Die Oberfläche von Magnetplatten kann beispielsweise durch Verunreinigungen beschädigt werden. Dadurch werden u.U. einzelne Sektoren oder Spuren physikalisch zerstört, was sich auch durch neues Formatieren nicht beseitigen läßt. Mit Hilfe von Reclaim können Platten vollständig geprüft und schadhafte Sektoren (d.h. Blöcke) ausgeblendet werden. Die fehlerhaften Blöcke werden in einem speziellen File zusammengefaßt (Benutzer 15, Attribut R/O). Selbstverständlich ist dies nicht möglich, wenn auch in den System- oder Directory-Spuren Fehler auftreten. Insbesondere für Festplatten (Winchester) oder Wechselplatten zahlt sich ein solches Dienstprogramm schnell aus.

Der Aufruf erfolgt im einfachsten Fall mit

```
A>RECLAIM↵
```

Anschließend wird das folgende Menü angezeigt:

```
RECLAIM Version 2.0
Copyright © 1980 Lifeboat Associates

Options available:
  E - EXAMINE a disk
  R - RESTART a suspended RECLAIM operation
  X - EXIT RECLAIM
 ^C - CONTROL-C interrupt processing
```

Im Fall der Funktion E fragt das Programm zunächst nach dem Laufwerk, in dem sich die fehlerhafte Diskette befindet. Danach muß zwischen den Funktionen

R	Nur-Lese Test (read-only)
D	Zerstörendes Lesen und anschließendes Schreiben (destructive write)
N	Nicht-zerstörendes Lesen/Schreiben/Lesen

gewählt werden.

Die Zeit für den Test ist nicht allein von der gewählten Funktion, sondern auch von der Magnetplatte und dem BIOS Ihres Personal Computers abhängig. Deshalb ist die Möglichkeit einer Unterbrechung des Testlaufs durch ^C besonders interessant. Sollten bereits Fehler in den System- oder Directory-Spuren aufreten, ist ein weiteres Testen ohnehin nicht sinnvoll, da i.a. in diesen Spuren alle Sektoren benötigt werden. Nach der Unterbrechung teilt Reclaim mit, wieviel Blöcke bisher als fehlerhaft erkannt wurden. Anschließend kann der Testlauf fortgesetzt werden.

Reclaim bietet nicht die Möglichkeit des Tests oder der Anzeige einzelner Sektoren oder Spuren. Dazu können Programme wie z.B. ZAP oder Disk Inspector eingesetzt werden.

Spezifikationen:

Programmname:	RECLAIM
Aufgabe:	Diagnose von Disketten und Festplatten
Hersteller:	Lifeboat Associates
Betriebssytem:	CP/M
Prozessor:	8080, 8085, Z80
Min. Speicher:	20 kByte
Anpassung:	nicht erforderlich
Bedienerführung:	einfache Menütechnik

Disk Inspector

Mit Hilfe des Dienstprogramms Disk Inspector können recht einfach Sektoren einer Diskette auf der physikalischen (Bit-) Ebene angezeigt und editiert werden. Wesentlich ist die Eigenschaft des Programms, daß zur gleichen Zeit auf dem Bildschirm 2 Sektoren mit je 128 Bytes aus dem gleichen File oder aus unterschiedlichen Dateien angezeigt werden können. Zusätzlich können diese Sektoren auch verglichen oder kopiert werden. Auf der einen Seite ist die Bedienerführung sehr einfach gehalten: Die Befehle werden in der ersten Zeile des Bildschirms angezeigt. Erläuterungen können allerdings durch die Help-Funktion aufgerufen werden. Andererseits ist das Editieren durch die Cursorsteuerung sehr einfach und schnell. Wesentliche Einschränkung für einige Benutzer ist lediglich die Tatsache, daß der Z80 als Prozessor in Ihrem Personal Computer vorausgesetzt wird.

Spezifikationen:

Programmname:	Disk Inspector
Aufgabe:	Anzeige und Editieren von Sektoren einer Diskette oder Festplatte
Hersteller:	Overbeek Enterprises
Betriebssytem:	CP/M
Prozessor:	Z80
Min. Speicher:	20 kByte
Anpassung:	Anpassung des Bildschirms (Cursorsteuerung) ist erforderlich; wird jedoch unterstützt
Bedienerführung:	sehr einfaches Menü

ZAP-80

Ähnlich wie der Disk Inspector gestattet das Programm ZAP-80 einen "Blick hinter die Kulisse" auf Ihren Disketten. Auch hier können einzelne Sektoren, Blöcke oder Dateien nicht nur angezeigt, sondern auch nach einer Modifikation zurückgespeichert werden. Diese werden alternativ über ihre physikalische Adresse (Spur und Sektor) oder über ihren Dateinamen angewählt. Dabei ist sogar (zumindest durch Angabe der physikalischen Adresse) die Bearbeitung von Disketten möglich, die vom BIOS Ihres Personal Computers nicht bearbeitet werden konnten.
Die Bedienung erfolgt menügesteuert. Im ersten Menü werden auch die aktuellen Disketten-Parameter angezeigt. Diese können bei Bedarf geändert werden. Vom ersten Menü können vier weitere Menüs angewählt werden:

Hauptmenü
File-Menü
Sonstige Befehle
Spezielle Funktionen

Vom ersten Menü können auch die Steuerzeichen eingegeben werden, die für die Anpassung des Bildschirms (Cursorsteuerung) erforderlich sind.

Spezifikationen:

Programmname:	ZAP-80
Aufgabe:	Anzeige und Editieren von Sektoren einer Diskette oder Festplatte
Hersteller:	Phase Four Software Inc.
Betriebssytem:	CP/M
Prozessor:	8080, 8085, Z80
Min. Speicher:	k.A.
Anpassung:	Eine Anpassung des Terminals (Cursorsteuerung) ist erforderlich
Bedienerführung:	Menütechnik

Die hier aufgeführten Programme sind nur Beispiele. Allerdings muß auch gesagt werden, daß solche Programme fast ausschließlich aus dem englischen Sprachbereich kommen (Dialog und Manuale bedienen sich der englischen Fachsprache). Weiterhin ist das Angebot derartiger Programme für 16-Bit-Systeme noch sehr gering. Die Adressen der Hersteller sind im Anhang Q enthalten.

KOMMUNIKATION ZWISCHEN CP/M-SYSTEMEN

Die serielle Verbindung von zwei oder mehr Personal Computern erhält eine rasch wachsende Bedeutung. Beispielsweise können auf diese Weise die Schwierigkeiten sehr einfach umgangen werden, die durch die große Vielfalt der Disketten-Aufzeichnungsformate - insbesondere bei den Mini-Disketten (5 1/4 Zoll) – entstehen. Bewährt hat sich unter CP/M beispielsweise das Programm ASCOM (Asynchronous Communication Protocol-Program), für das sowohl eine 8-Bit- als auch eine 16-Bit-Version angeboten wird. Der Komfort und die Flexibilität von ASCOM sind beachtlich (über 50 Befehle).

Selbstverständlich müssen solche Programme an die Hardware Ihres Personal Computers angepaßt werden. Dies bedeutet, daß ein entsprechender Treiber geschrieben werden muß, falls dieser nicht bereits enthalten ist.

Das mit ASCOM gelieferte (englische) Manual (190 Seiten) beschreibt nicht nur die Befehle, sondern behandelt auch ausführlich die für eine Anpassung gegebenenfalls notwendigen Arbeiten.

Spezifikationen:

Programmname:	ASCOM-80, ASCOM-86
Aufgabe:	Serielle Kommunikation zwischen Personal Computern
Hersteller:	Dynamic Microprocessor Associates
Betriebssytem:	CP/M, CP/M-86, MS-DOS, PC-DOS
Prozessor:	8080, 8085, Z80, 8086, 8088
Anpassung:	Treiber
Bedienerführung:	Einfache Befehlssprache und Menütechnik

ANPASSUNG VON CP/M-SOFTWARE AN PERSONAL COMPUTER

Ein wesentlicher Grund für den Erfolg der CP/M-Familie ist die Möglichkeit, Programme auf der Ebene der Objektprogramme, d.h. der Maschinenbefehle, zwischen sehr unterschiedlichen Rechnern austauschen zu können. Dadurch können für sehr leistungsfähige Programme wie Com-

piler, Datenbank-Systeme oder Grafik-Pakete sehr große Stückzahlen verkauft werden. Dies ist wiederum die entscheidende Voraussetzung für eine (für den Anwender) interessante Preisgestaltung. Einige Compiler sind heute bereits unter DM 150,- erhältlich!

Obwohl CP/M eine optimale Basis für portable Programme bietet, sind trotzdem Anpassungsarbeiten bei einigen Produkten erforderlich. Die Ursache dafür sind fehlende Standards insbesondere für Bildschirm-Terminals und Drucker, deren „Intelligenz“ (Leistung) immer weiter steigt. Gute Pakete bieten bereits integrierte (interaktive) Installationsprogramme an. Dadurch wird die Eingabe der jeweiligen Sequenzen von Steuerzeichen z.B. für die Cursorsteuerung sehr erleichtert. Zu berücksichtigen ist dabei, daß ein Teil der Angaben unbedingt erforderlich ist. Andere wiederum dienen lediglich einer Erhöhung der Arbeitsgeschwindigkeit.

In einigen Fällen, wie z.B. der Verwendung von speziellen Schnittstellen Ihres Personal Computers oder von Grafik-Eigenschaften, sind für die Anpassungsarbeiten fundierte Programmiererfahrungen und unter Umständen sogar Hardwarekenntnisse erforderlich.

BENUTZERGRUPPEN UND „PUBLIC-DOMAIN“-SOFTWARE

Dank der bereits erwähnten starken Verbreitung der CP/M-Familie und der Möglichkeit des Programmaustauschs haben sich Benutzergruppen gebildet. Dies sind vor allem die CPMUG (CP/M Users Group) und der European Users Club (siehe auch Anhang Q). Eine der wesentlichen Ergebnisse der Arbeit dieser Gruppen ist der Aufbau einer sehr großen Bibliothek von Programmen. Zur Zeit sind über 100 Programmdisketten verfügbar. Diese Programme werden von den Mitgliedern der Benutzergruppe kostenlos zur Verfügung gestellt (public domain). Von den Benutzergruppen werden dann nur die Kosten für Material und Versand (DM 25,- pro 8-Zoll-Diskette) berechnet. Eine Übersicht über die verfügbaren Programme ist in einem Katalog enthalten. Der Inhalt neuer Disketten der CPMUG wird in der Zeitschrift „Lifelines“ von Lifeboat veröffentlicht.

MASSNAHMEN NACH EINEM SYSTEMFEHLER

Fehlbedienung durch Operator:

1. Prüfen der mechanischen Voraussetzungen:
   - Sind alle Schalterstellungen korrekt (systematische und vollständige Überprüfung unbedingt erforderlich)?
   - Sind die Sicherungen in Ordnung?
   - Sind alle Kabelverbindungen in Ordnung?

2. Wurde ein korrekter Befehl eingegeben?
   – Abschalten des Systems, Einschalten des Systems.
   – Erneute Eingabe des gewünschten Befehls.

Fehlerquelle Diskette:

3. Benutzung einer neuen Diskette (häufig wurde die benutzte Diskette durch falsche Bedienung zerstört, was u.U. zu einem falschen Systemverhalten führt).
   – Benutzung einer „Back-Up"-Diskette. Auf keinen Fall sollte die bis dahin benutzte Diskette bzw. das Programm verwendet werden.
   – Wenn keine „Back-Up"-Diskette existiert, sollte direkt eine solche erstellt werden.

Verdacht auf Softwarefehler:

4. Überprüfung auf Verwendung des korrekten Programms:
   – richtige CP/M-Version
   – der richtige Rechner/Interpreter zum verwendeten Anwendungsprogramm (z.B. kann für ein CBASIC-Programm CBASIC-Version 2 notwendig sein).
   – das richtige Anwendungsprogramm.

Viele Anwendungsprogramme, besonders Textverarbeitungsprogramme, müssen auf eine spezielle Rechnerkonfiguration einschließlich Drucker und Bildschirmterminal angepaßt sein. Ist dies nicht der Fall, besteht die Möglichkeit, daß z.B. einige Tasten am Bildschirmterminal und der Drucker nicht die gewünschte Funktion ausführen.

Verdacht auf Hardwarefehler:

5. Nochmalige Überprüfung der Elektronik und Mechanik
   – Herausnehmen der Platinen, falls notwendig, Reinigen der Verbindungen und richtiges Einstecken in die vorgesehenen Fassungen.
   – Wenn die Fehlfunktionen eindeutig einer Platine zugeordnet werden können, empfiehlt sich das Herausnehmen der ICs, Reinigen der Verbindungen und richtiges Einstecken der ICs in die Fassungen.

6. Es sollte systematisch versucht werden, die fehlerhafte Stelle der Rechnerkonfiguration zu identifizieren. Dies kann durch definiertes Zu- und Abschalten einzelner Komponenten schnell ermittelt werden. Übereilte Fehlerzuordnungen sind jedoch zu vermeiden (z.B. durch eine Gegenprüfung).

7. Zur weiteren Vermeidung von Hardwarefehlern sind von nun an alle in diesem Buch und in den Handbüchern des Rechners erläuterten Präventivmaßnahmen zu beachten. Hierzu gehört besonders
   - das Einhalten einer hohen Benutzerdisziplin (siehe 7.1)
   - Erfüllung selbst überlegter Nutzungsregeln (keine Ausnahmen von den Regeln!).

Kapitel 8

Entwicklungstendenzen

GESCHICHTLICHE ENTWICKLUNG VON CP/M

Das Betriebssystem CP/M wurde von Gary Kildall entwickelt. Gary Kildall schrieb als Angestellter der Firma Intel Corporation den ersten Compiler für die von Intel vertriebene höhere Programmiersprache PL/M. 1974 entwickelte er unter dem Namen CP/M die erste Version eines Dateiverwaltungssystems, das zu dieser Zeit lediglich zur Unterstützung eines residenten PL/M-Compilers gedacht war.

CP/M wurde 1975 zum erstenmal kommerziell angeboten, blieb jedoch fast über ein Jahr nach der Lizenzvergabe relativ unbeachtet. Während dieser Zeit wurden die ersten Versionen des Editors (ED), des Assemblers (ASM) und des Debuggers (DDT) entwickelt. Die erste größere kommerzielle Anwendung dieses Betriebssystems begann nach der Vergabe einer Vetriebslizenz der CP/M-Version 1.3 an die Firma IMSAI (existiert heute nicht mehr), die CP/M kurze Zeit später mit erweiterten Funktionen als IMDOS-Betriebssystem anbot. CP/M selbst wurde zur Version 2.2 weiterentwickelt, die in dieser Konfiguration auch die Verwaltung großer externer Speicher wie Festkopfplatten erlaubt. MP/M wurde speziell für die Unterstützung von Multi-User- und Multi-Tasking-Aufgaben entwickelt. Heute wird durch CP/M-Plus für die 8-Bit- und Concurrent CP/M-86 für die 16-Bit-Welt eine neue Generation der CP/M-Familie angeboten. Erwähnenswert ist dabei, daß die Version 3.1 von Concurrent CP/M nicht nur Multi-Tasking bietet, sondern auch einen Mehrbenutzer-Betrieb zuläßt. Weiterhin können unter dieser Version auch IBM PC-DOS-Programme eingesetzt werden.

Ergänzt wird die CP/M-Familie durch Netzwerk-Software. Für die 8-Bit-Systeme wird bereits seit ein paar Jahren von Digital Research CP/NET angeboten. Dieses setzt im Master (von Digital Research „server" genannt) als Betriebssystem MP/M und in den Slaves (Satelliten, von Digital Research „requester" genannt) CP/M oder MP/M voraus. Für 16-Bit-Systeme wird heute das mit CP/NET verträgliche SoftNet angeboten. Dadurch kann auf dieser Basis auch ein Netzwerk aufgebaut werden, das 8- und 16-Bit-Systeme enthält.

Schließlich muß hier noch „Personal CP/M" erwähnt werden. Dieses ist

ein (ROM-) residentes CP/M, das keine Platte voraussetzt. Es kann beispielsweise sehr gut in Netzwerken eingesetzt werden.

Heute zählen die Betriebssysteme der CP/M-Familie zu den meistbenutzten Betriebssystemen für Mikrocomputer. Trotz einiger Schwächen von CP/M 2.2 und CP/M-86, die vornehmlich jedoch von Systemnutzern und -entwicklern angeführt werden, die CP/M mit Betriebssystemen von Mini- und Großcomputern vergleichen, ist es z.Z. nicht wegzudenken und gilt als de-facto-Standard vieler Mikrocomputer-Anwender.

CP/M UND ANDERE BETRIEBSSYSTEME

Die Entwicklung leistungsfähiger Betriebssysteme ist immer mit hohen Investitionen verbunden. Aus diesem Grunde sind umfangreiche Mehrbenutzer-Betriebssysteme bisher nur für wenige Mini- und Großcomputer realisiert. Die komplexe Struktur von Betriebssystemen weisen Mehrbenutzer-Systeme mit leistungsfähigen Zuteilungs- und Sicherungsstrategien auf. Welches der realisierten Systeme das leistungsfähigste ist, läßt sich abschließend nicht festlegen, da dies nur am Aufgaben- und Anwendungssystem effektiv meßbar ist. Im Bereich der Minicomputer und hier insbesondere bei den 16-Bit-Minicomputern hat UNIX eine weite Verbreitung erreicht.

Einige Versuche sind auch bereits gemacht worden, UNIX auf 16-Bit-Mikrocomputern und 8-Bit-Mikrocomputern (CROMIX von Cromemco und XENIX von Microsoft) zu implementieren. Die notwendigen Investitionen, hier eine volle Verträglichkeit mit UNIX zu erreichen, sind jedoch sehr umfangreich und die Wahrscheinlichkeit eines vollständigen Erfolgs begrenzt.

Auf jeden Fall muß hier noch in diesem Zusammenhang das Betriebssystem MS-DOS von Microsoft (für 16-Bit-Systeme) erwähnt werden, das durch seine Implementierung für den IBM PC (unter dem Namen PC-DOS) eine große Verbreitung erreicht hat. Allerdings muß auch gesagt werden, daß PC-DOS ebenso wie MS-DOS im Gegensatz zu MP/M und Concurrent CP/M kein Multi-Tasking (Parallelverarbeitung) bieten.

Der bedeutendste Vorteil von CP/M ist die erreichte Verträglichkeit zwischen Dateien und Programmen für die unterschiedlichsten Hardwaresysteme. Der Austausch von CP/M-verträglichen Programmen ist damit jederzeit gewährleistet. Aus diesem Grunde wird CP/M mit einiger Sicherheit noch lange Zeit als Systemstandard betrachtet werden – zumindest so lange, wie die Basisprozessoren 8080, Z80, 8085, 8086 und 8088 noch in hoher Stückzahl für Personal Computer benutzt werden.

ENTWICKLUNG

Verbesserungen (und Korrekturen) an existierenden Quellprogrammen können jederzeit durchgeführt werden. Aus diesem Grunde ist zu erwar-

ten, daß sowohl MP/M als auch CP/M ständig verbessert werden. Dies ist an Produkten wie CP/M-Plus, Concurrent CP/M und SoftNet sehr deutlich zu erkennen. Hierbei kann davon ausgegangen werden, daß diese neuen Systemversionen verträglich mit den älteren Versionen sind. Das bedeutet, daß der Leser dieses Handbuchs auch mit dem Grundbefehlsvorrat neuerer Versionen von CP/M und MP/M genügend vertraut ist. Darüber hinaus sind auch die Grundfunktionen anderer Betriebssysteme vom Konzept her erfaßt und leicht adaptierbar.

SCHLUSS

Nach dem Durcharbeiten dieses Buches kann ein verfügbarer Personal Computer benutzt und für individuelle Zwecke eingesetzt werden. Insbesondere hierfür wurde das „CP/M-Handbuch" geschrieben.

Nach der Erfassung der wesentlichen Eigenschaften eines Programmsystems wie CP/M ist eine hohe Benutzerdisziplin der Schlüssel für eine fehlerfreie und problemlose Nutzung des Gesamtsystems. Werden alle aufgeführten Hinweise vollständig befolgt, so ist der Einsatz von CP/M nahezu problemlos. Besonders zu Beginn sollten alle Anweisungen ohne Ausnahme genauestens befolgt werden. Bei fortlaufender Erfahrung mit dem System ist der Anwender dann in der Lage, einige Regeln zu modifizieren oder wegzulassen. Hinweise und Regeln zur Behandlung und Nutzung kleinerer und mittlerer Computer und deren Peripherieeinheiten werden in „Mein erster Computer" (Rodnay Zaks, SYBEX-Verlag) behandelt.

Abhängig vom jeweiligen Wissensstand kann jedes Kapitel dieses Buches nach Bedarf erarbeitet werden. Mit Ausnahme des ersten Kapitels können aus jedem Kapitel ohne Schwierigkeiten die gewünschten Teilaspekte betrachtet werden, da im Anhang in Kurzform nochmals alle Funktionen zusammengefaßt sind.

Mit Nutzung eines Computersystems werden im Zeitablauf nach und nach alle Funktionen erfaßt und wirkungsvoll einsetzbar. Soll z.B. zunächst das System nur für die Texterfassung eingesetzt werden, so ist dies mit der Bearbeitung des Kapitels über den Editor ED sofort möglich. Weiterführende Programme zur Textbearbeitung und Textverarbeitung sowie andere kommerziell einsetzbare Programme sind zu einem späteren Zeitpunkt dann leicht ausführbar (siehe auch H. Glatzer, Einführung in die Textverarbeitung, SYBEX-Verlag).

Sind Grundkonzepte und Techniken dieses Buches einmal erfaßt, so ist ein hohes Maß an Fachkenntnis über den Einsatz von Systemsoftware auf Mikrocomputern vorhanden und eine schnelle Adaption anderer Programme und Betriebssysteme möglich.

Anhang A

Allgemeine CP/M-Fehlermeldungen

Fehlermeldungen können sowohl vom Betriebssystem als auch von Programmen ausgegeben werden. Die meisten vom Betriebssystem angezeigten Fehler führen zum Abbruch des Programms. Bei neueren Systemen (CP/M-Plus, MP/M und CCP/M) kann vom Programm aus entschieden werden, ob das Betriebssystem die Fehlermeldung ausgibt oder ob das Programm diese Aufgabe übernimmt (und somit die Bearbeitung fortsetzen oder definiert abbrechen kann).

Beim Standard-CP/M werden Fehlermeldungen in folgendem Format mitgeteilt:

```
BDOS ERR ON d:Fehlerursache
```

Der Buchstabe d identifiziert das Laufwerk, auf dem der Fehler erkannt wurde. Für das Wort Fehlerursache steht eine der folgenden Meldungen:

```
BAD SECTOR
SELECT
READ ONLY
FILE R/O
```

BAD SECTOR

Ein „bad sector“ (fehlerhafter Disketten- oder Plattensektor) wird angezeigt, wenn bei einer Lese- oder Schreiboperation ein physikalischer Fehler aufgetreten ist. So ein Fehler kann dadurch entstehen, daß Disketten noch nicht oder falsch formatiert wurden, daß sie mechanisch defekt sind oder daß sie aus irgendeinem Grund falsch beschrieben wurden.

Zur Quittierung dieser Fehlermeldung kann mit CTRL-C ein erneuter Systemstart eingegeben werden. Dadurch wird das laufende Programm oder die gerade durchgeführte Dateibearbeitung unterbrochen. Diese Fehlermeldung kann auch durch Drücken der RETURN-Taste quittiert werden. In diesem Fall wird die Fehlermeldung und damit der fehlerhafte Sektor ignoriert.

Das Ignorieren gemeldeter Fehler kann in einigen Fällen zu weiteren Zerstörungen auf der Diskette führen. Tritt der Fehler z.B. beim Beschreiben der Directory auf und wird ignoriert, so kann die Directory zerstört und damit jeder Zugriff auf irgendeine Datei dieser Diskette unmöglich werden. Auch beim Laden von Programmen sollten Lesefehler nicht ignoriert werden.

SELECT

Diese Fehlermeldung wird angezeigt, wenn ein nicht existierendes Laufwerk angesprochen wird. Die Angabe d weist auf das selektierte Laufwerk hin, für das der SELECT-Fehler gilt. Nach Betätigung einer beliebigen Taste wird ein Warmstart ausgelöst.

READ ONLY

Die Fehlermeldung „READ ONLY" wird angezeigt, wenn versucht wurde, auf ein Laufwerk zu schreiben, das einen logischen Schreibschutz erhalten hat. Der logische Schreibschutz kann von einem Programm oder vom STAT-Befehl eingeschaltet werden. Er entsteht aber auch durch Wechseln von Disketten ohne nachfolgenden Warmstart (CTRL-C).

Eine READ ONLY-Fehlermeldung kann durch das Drücken einer beliebigen Terminaltaste quittiert werden. Danach wird automatisch ein Systemstart erzeugt, der dann außerdem die betreffende Diskette wieder in den READ/WRITE-Betrieb schaltet.

FILE R/O

Wird versucht, auf eine Datei zu schreiben oder eine solche Datei zu löschen oder umzubenennen, bei der das Schreibschutz-Attribut (R/O) gesetzt ist, wird die Operation mit dieser Fehlermeldung abgebrochen.

Die Möglichkeiten der Anwendung von Dateiattributen wurde im Kapitel 2 beschrieben. Die FILE R/O-Fehlermeldung kann durch Drücken einer beliebigen Terminaltaste quittiert werden. Danach muß jedoch, wenn nötig, mit dem STAT-Befehl oder einem anderen Programm das $R/O-Attribut durch ein $R/W-Attribut ersetzt werden.

Bei den neueren Systemen (CP/M-Plus, MP/M und CCP/M) ist der Schutz der Dateien vor fehlerhaft arbeitenden Programmen wesentlich verbessert worden, was die Anzahl der möglichen Fehler natürlich erhöht. Die Fehlermeldung ist im Prinzip folgendermaßen aufgebaut (kann je nach System etwas variieren):

```
Bdos Err on d:   Fehlerursache
Function NNN     File: Dateiname
```

Dabei ist d das Laufwerk, NNN die BDOS-Funktionsnummer (siehe Kapitel 5) und „Dateiname“ der Name der Datei, bei der der Fehler auftrat (wenn der Befehl Datei-bezogen war). Für „Fehlerursache“ können folgende Texte stehen:

Bad Sector
oder
Disk I/O

Ein physikalischer Fehler beim Lesen oder Schreiben der Platte wurde festgestellt.

Select
oder
Invalid Drive

Ein nicht existierendes Laufwerk wurde angewählt.

Read/Only Disk

Es wurde versucht, auf ein Laufwerk zu schreiben, das einen logischen oder physikalischen Schreibschutz hat. Der logische Schreibschutz kann bei CP/M-Plus durch einen Warmstart und bei MP/M sowie CCP/M durch den Befehl DSKRESET beseitigt werden.

Read/Only File

Es wurde versucht, auf eine schreibgeschützte Datei (Read-Only-Attribut) zu schreiben bzw. zu löschen oder umzubenennen. Bei MP/M sind diese beiden Fehlermeldungen zusammengefaßt zu R/O.

Passwort error

Der Paßwort-Schutz einer geschützten Datei wurde verletzt.

File already exists

Es wurde versucht, eine Datei anzulegen, die unter dem gleichen Namen im gleichen Benutzerbereich schon auf der angewählten Platte existiert.

Illegal ? in filename

Bei einer Operation, die keine Fragezeichen im Dateinamen zuläßt, wurde im FCB (File Control Block) ein ? gefunden.

Die folgenden Fehlermeldungen treten nur bei MP/M und CCP/M auf:

Close Checksum Error

Bei Schließen einer Datei ist festgestellt worden, daß der FCB vom Programm geändert wurde.

Open File Limit Exceeded

Die Anzahl der geöffneten Dateien übersteigt die mit „GENSYS" vereinbarte Zahl.

No Room in System Lock List

Die Anzahl der Sperreinträge in die „System Lock List" übersteigt die mit GENSYS vereinbarten Werte.

File Opened in Read/Only Mode
und
File Currently Open

Diese Fehlermeldungen betreffen den Schutz der Dateien vor konkurrierenden Zugriffen. Näheres ist den entsprechenden Manualen zu entnehmen.

Eine andere wichtige Gruppe von Fehlermeldungen kann beim Laden von Programmen auftreten:

BAD LOAD
oder
Prg ld err
oder
?Not Enough Memory

Das Programm ist zu lang, um vollständig geladen werden zu können.

Abs TPA not free

Diese MP/M-80 Fehlermeldung zeigt an, daß kein absolutes Speichersegment (ab Adresse 0) zum Laden der .COM-Datei frei ist.

Reloc seg not free

Diese MP/M-80 Fehlermeldung zeigt an, daß kein Speichersegment zum Laden der .PRL-Datei frei ist.

Die Fehlermeldungen der Programme sind natürlich sehr vielfältig. Als Beispiel sollen hier die Meldungen des PIP-Befehls beschrieben werden:

(USER) ABORTED

Die PIP-Operation wurde durch Eingabe von CTRL-C abgebrochen.

BAD PARAMETER

Es wurde ein Fehler im Parameter-Feld festgestellt.

CANNOT CLOSE

Beim Schließen der Zieldatei ist ein Fehler aufgetreten.

DISK READ ERROR

Beim Standard-CP/M erfolgt diese Fehlermeldung, wenn PIP versucht hat, hinter dem Ende einer Datei weiterzulesen. Bei neueren Systemen wird diese Fehlermeldung mit Zusätzen differenziert.

DISK WRITE ERROR

Beim Standard-CP/M erfolgt diese Fehlermeldung, wenn beim Schreiben einer Datei entweder der Datenbereich der Platte voll wird oder aber der Directory-Bereich. Bei neueren Systemen gibt eine weitere Angabe Aufschluß über die genaue Fehlerursache.

FILE NOT FOUND

Die angegebene Quell-Datei ist nicht gefunden worden.

HEX RECORD CHECKSUM

Beim Kopieren von Dateien im Intel-Hex-Format (I- oder H-Option) ist ein Prüfsummen-Fehler aufgetreten.

INVALID DESTINATION

Die Angabe des Kopier-Zieles war fehlerhaft.

INVALID DIGIT

Ein Zeichen, das nicht im Hexadezimalbereich (0...9, A..F) liegt, wurde beim Kopieren einer HEX-Datei (H- oder I-Option) erkannt.

INVALID DISK SELECT

Es wurde versucht, ein nicht existierendes Laufwerk anzuwählen. (Beim Standard-CP/M gibt es diese Meldung nicht, da bereits das BDOS auf den Fehler reagiert.)

INVALID FILENAME

Ein angegebener Dateiname entspricht nicht den Vorschriften.

INVALID FORMAT

Das Befehlsformat entspricht nicht den Vorschriften. (Z.B. Versuch, eine Dateigruppe auf das gleiche Laufwerk zu kopieren.)

INVALID PASSWORD

Das angegebene Paßwort ist falsch, oder die Datei hat keinen Paßwort-Schutz (nicht bei Standard-CP/M).

INVALID SEPARATOR

Zwischen zwei Dateinamen wurde ein falsches Trennzeichen erkannt.

INVALID SOURCE

Die Angabe der Kopier-Quelle war fehlerhaft.

INVALID USER NUMBER

Die mit der G-Option angegebene Benutzernummer lag nicht im Bereich 0...15.

NO DIRECTORY SPACE

Die Directory des Ziellaufwerkes ist voll.

QUIT NOT FOUND

Die mit der Q-Option vereinbarte Stopp-Zeichenkette ist nicht gefunden worden.

REQUIRES CP/M 3

Die Version des Betriebssystems stimmt nicht mit der erforderlichen Version (hier CP/M-Plus) überein.

START NOT FOUND

Die mit der S-Option vereinbarte Start-Zeichenkette ist nicht gefunden worden.

VERIFY ERROR

Beim anschließenden Prüf-Lesen einer kopierten Datei ist eine Abweichung festgestellt worden. Diese Fehlermeldung kann nur auftreten, wenn mit der Verify-Option [V] gearbeitet wurde.

Anhang **B**

Hexadezimal-Konversionstabelle

HEX	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	00	000
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	0	0
1	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	256	4096
2	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	512	8192
3	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	768	12288
4	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	1024	16384
5	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	1280	20480
6	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	1536	24576
7	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	1792	28672
8	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	2048	32768
9	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	2304	36864
A	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	2560	40960
B	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	2816	45056
C	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	3072	49152
D	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	3328	53248
E	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	3584	57344
F	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	3840	61440

Anhang C

ASCII-Konversionstabelle

BIT NUMBERS

							b7 →	0	0	0	0	1	1	1	1
							b6 →	0	0	1	1	0	0	1	1
							b5 →	0	1	0	1	0	1	0	1
b7	b6	b5	b4	b3	b2	b1	HEX 1 / HEX 0	0	1	2	3	4	5	6	7
			0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
			0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
			0	0	1	0	2	STX	DC2	"	2	B	R	b	r
			0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
			0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
			0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
			0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
			0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
			1	0	0	0	8	BS	CAN	(	8	H	X	h	x
			1	0	0	1	9	HT	EM	)	9	I	Y	i	y
			1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
			1	0	1	1	11	VT	ESC	+	;	K	[	k	{
			1	1	0	0	12	FF	FS	,	<	L	\	l	¦
			1	1	0	1	13	CR	GS	-	=	M	]	m	}
			1	1	1	0	14	SO	RS	.	>	N	^	n	~
			1	1	1	1	15	SI	US	/	?	O	_	o	DEL

NUL Null
SOH Beginn des Kopfes (Start of Heading)
STX Beginn des Textes (Start of Text)
ETX Ende des Textes (End of Text)
EOT Ende der Übertragung (End of Transmission)
ENQ Anfrage (Enquing)
ACK Bestätigung (Acknowledge)
BEL Klingel (Bell)
BS Zurücksetzen (Backspace)
HT Horizontaler Tabulator (Horizontal Tabulation)
LF Zeilenvorschub (Line Feed)
VT Vertikaler Tabulator (Vertical Tabulation)
FF Format-Vorschub (Form Feed)
CR Wagenrücklauf/Zeilenwechsel (Carriage Return)
SO Rückschaltung (Shift Out)
SI Dauerumschaltung (Shift In)
DLE Datenverbindungs-Umschaltung (Data Link Escape)
DC Gerätesteuerung (Device Control)
NAK Negativ-Bestätigung (Negative Acknowledge)
SYN Synchronisations-Leerlauf (Synchronous Idle)
ETB Ende des Übertragungsblocks (End of Transmission Block)
CAN Annullieren (Cancel)
EM Datenträgerende (End of Medium)
SUB Ersetzen (Substitute)
ESC Umschaltung (Escape)
FS Dateitrennzeichen (File Separator)
GS Gruppentrennzeichen (Group Separator)
RS Satztrennzeichen (Record Separator)
US Einheiten-Trennzeichen (Unit Separator)
SP Leerzeichen (Space/Blank)
DEL Löschzeichen (Delete)

Anhang D

ED-Steuerbefehle

Taste	Bedeutung
CTRL-C (^C)	System Restart (Warmstart), Wiederholung System Prompt.
CTRL-E (^E)	Positioniert den Cursor zur nächsten Zeile für die Fortsetzung einer Befehlszeile (Befehlszeile wird noch nicht abgeschlossen).
CTRL-H (^H)	Setzt Cursor um eine Position zurück und löscht das zugehörige Zeichen. Entspricht BACKSPACE.
CTRL-I (^I)	Weiterschieben des Cursors um eine Tabulatorbreite (7 Spalten). Entspricht TAB.
CTRL-J (^J)	Auslösen eines RETURN.
CTRL-M (^M)	Auslösen eines RETURN.
CTRL-L (^L)	Symbol für die Austauschsequenz eines „RETURN“-Zeichens in Zeichenketten (z.B. in Such- und Austauschbefehlen).
CTRL-R (^R)	Wiederholte Anzeige der aktuellen Befehlszeile (bereinigt um korrigierte Zeichensymbole).
CTRL-U (^U)	Löschen der aktuellen Befehlszeile.
CTRL-X (^X)	Löschen der aktuellen Befehlszeile und Positionieren des Cursors an den Beginn der Zeile.
CTRL-D (^D)	Wegschalten des aktuellen Programms in den Hintergrundbetrieb (MP/M).
RETURN (↵)	Übernahme (Ausführen) der aktuellen Befehlszeile bzw. Erzeugen eines Carriage Return in einer Textzeile.
CTRL-Z (^Z)	Abschluß des Einfügebetriebs (I-Befehls), logische Trennung von Textteilen in Such- und Austauschbefehlen oder Positionieren eines EOF-Zeichens am Ende einer Textdatei.

CTRL-P (^P)	Zuschalten bzw. Wegschalten des Druckers (Anzeige über Displayterminal wird ebenfalls über den Drucker ausgegeben).
CTRL-S (^S)	Stoppen einer Anzeige auf dem Displayterminal (wird durch Drücken von CTRL-Q, bei CP/M-80 auch einer beliebigen Taste wieder fortgeführt).
CTRL-Q (^Q)	Weiterlauf der Konsol-Ausgabe nach dem Stoppen mit CTRL-S.

Anhang E
ED-Befehle

Befehl	**Bedeutung**
nA	Laden (append) von n Zeilen (oder 1, wenn n nicht angegeben) aus der Quelldatei in den Editpuffer. Ein # für n erzeugt das Laden von 65535 Zeilen (Füllen des gesamten Puffers), und 0 für n lädt soviel Zeilen, bis die Hälfte des Puffers gefüllt ist (Anzahl der geladenen Zeilen ist von der Größe des Arbeitsspeichers und vom System abhängig).
+/−nB	Positioniert den CP an den Beginn (+B) oder das Ende (−B) des Puffers.
+/−nC	Positioniert den CP um (+n) Zeichen vorwärts oder (−n) Zeichen rückwärts. Ein Carriage Return wird als zwei Zeichen gezählt (RETURN und LINE FEED).
+/−nD	Löscht n Zeichen vorwärts (+n) oder n Zeichen rückwärts (−n) einschließlich der Carriage Return-Zeichen. Ist keine Zahl für n angegeben, so wird das Zeichen neben dem CP gelöscht.
E	Beendet den Editierbetrieb. Der E-Befehl sichert den gepufferten Text und den Rest der Quelldatei in einer Übergangsdatei, benennt die Quelldatei in die entsprechende BAK-Datei um und belegt die Übergangsdatei mit dem Namen der ursprünglichen Quelldatei. Danach wird der Editierbetrieb abgebrochen, und das System meldet sich wieder mit dem Bereitschaftszeichen.

WICHTIG: In allen ED-Befehlen, wo mit n eine bestimmte Anzahl von Operationen angegeben wird (immer), kann mit # die höchstmögliche Zahl (65535) angegeben werden. Auf diese Weise werden die Befehlseingaben häufig kürzer.

nFstring {^Z / ↵} — sucht die n-te durch die Zeichenkette angegebene Zeichenfolge (wenn n nicht angegeben wird, wird die erste Zeichenkette gesucht). Mit F wird beginnend vom CP der gesamte Textpuffer abgesucht und nach dem Auffinden der Zeichenkette der CP auf das folgende Zeichen gesetzt. Ein folgendes ^Z erlaubt das zusätzliche Anfügen weiterer Befehle. Wird dies nicht gewünscht, genügt der Abschluß mit RETURN.

H — Beendet den Editierbetrieb wie der E-Befehl mit dem Unterschied, daß die so erzeugte neue Quelldatei erneut aufgerufen wird und damit der Editierbetrieb fortgesetzt werden kann.

I↵ — Einfügen einer neuen Textzeile nach dem CP. Der CP wird an das Ende der zuletzt eingegebenen Zeile gesetzt.

- Wenn der I-Befehl als Großbuchstabe I eingegeben wird, werden alle folgenden Zeichen auch als Großbuchstaben interpretiert.
- Wird der I-Befehl als Kleinbuchstabe i eingegeben, werden alle folgenden Zeichen entsprechend der Eingabe als Groß- bzw. Kleinbuchstaben interpretiert.

Itext^Z — Fügt die eingegebenen Zeichen beginnend von der aktuellen Cursorposition an ein und positioniert den CP hinter das zuletzt eingegebene Zeichen.

nJstring1^Zstring2^Zstring3 {^Z / ↵} — Nebeneinanderstellen (Juxtapose) von Zeichenketten, indem zuerst die erste Zeichenkette aufgesucht wird, die zweite Zeichenkette an die erste angefügt und alle Zeichen bis zum Erkennen der dritten Zeichenkette gelöscht werden. Der CP wird danach auf den Beginn der dritten Zeichenkette positioniert.

+/−nK — Löscht („Kill") die folgenden (+) bzw. die vorgehenden (−) n Zeilen und positioniert den CP direkt hinter die gelöschten Zeilen. Der CP wird hierbei mitgezählt.

+/−nL — Positioniert den CP um n Zeilen vorwärts (+) oder um n Zeilen rückwärts (−). Ist n = 0, so wird der CP an den Beginn der aktuellen Zeile gesetzt.

nMstring {^Z oder ↵} Führt den mit dem String angegebenen ED-Befehl n-mal (n>1) in derselben Weise aus. Wenn n gleich 1 oder 0 ist, wird der ED-Befehl so oft ausgeführt, bis das Ende der Datei erreicht ist oder ein Fehler erkannt wird.

nNtext {^Z oder ↵} Aufsuchen der n-ten mit Text angegebenen Zeichenfolge im Textpuffer und, falls dort nicht vorhanden, auch in der Quelldatei. Hierbei werden so viele Zeilen geladen (append), bis die gesuchte Zeichenfolge gefunden ist. Danach wird der CP an das Ende des gesuchten Textes gesetzt (bei der Angabe von ^Z können weiter Befehle angefügt werden).

+/−nT (Type) Wenn n nicht oder mit 1 angegeben ist, werden alle Zeichen in der aktuellen Zeile angezeigt. Wenn n mit Null angegeben ist, werden alle Zeichen vom Beginn der aktuellen Zeile bis zum CP angezeigt. Ist n positiv (+), werden die folgenden n Zeilen (einschließlich der aktuellen Zeile) angezeigt. Ist n negativ (−), so werden die n vorangehenden Zeilen ausschließlich der aktuellen Zeile angezeigt. Der CP wird hierbei nicht weiterbewegt. Mit „B#T“ kann der gesamte Textpufferinhalt angezeigt werden.

+/−U Übersetzt alle folgenden eingegebenen Kleinbuchstaben (lower case) nach der Eingabe von +U in Großbuchstaben. Diese Umsetzung wird beendet, wenn −U eingegeben wird.

V Einschalten der Anzeige von Zeilennummern der Textzeilen im Textpuffer (diese Nummern werden in der Datei nicht mitgeführt).

0V zeigt die verfügbare und die gesamte Größe des Textpuffers in Bytes (dezimal) an. So bedeutet z.B. die Anzeige 27648/28832, daß 27648 Bytes im Textpuffer noch nicht belegt sind und der Textpuffer eine Gesamtgröße von 28832 Bytes hat.

nW Schreiben von n Zeilen beginnend vom CP aus in die entsprechende Übergangsdatei der Kennung .$$$. Ist für n keine Zahl angegeben, wird nur die aktuelle Zeile weggeschrieben.

nX	Kopiert die folgenden n Textzeilen in die Datei X$$$$$$$.LIB (die Originalzeilen werden nicht gelöscht). Die Zeilen können mit dem R-Befehl wieder geladen werden. Wenn n gleich Null ist, wird mit diesem Befehl die Datei X$$$$$$$.LIB gelöscht.
nZ	Unterbricht laufende ED-Operationen um n Zeittakte (etwa n Sekunden).
+/−n	Bewirkt einen +/−nLT-Befehl.
n:	Positioniert den CP an den Beginn der Zeile n.
n1::n2	Spezifiziert einen Bereich von Zeilennummern beginnend mit n1 und endend mit n2. Fehlt entweder n1 oder n2, wird dieses Argument durch die aktuelle Zeile ersetzt.

Anhang F

PIP-Gerätenamen

Logische Gerätenamen

CON: Für Console oder ein Terminal, das Tastatur und Bildschirm besitzt (Eingabe/Ausgabe)

AXI: Beliebiges Eingabe-Gerät (bei CP/M-80 RDR:)

AXO: Beliebiges Ausgabe-Gerät (bei CP/M-80 PUN:)

LST: Für „Listeinheit" wie Matrixdrucker (nur Ausgabe)

Physikalische Gerätenamen bei CP/M-80

TTY: für Konsole bzw. Terminal: Leser, Stanzer oder LIST-Einheit (Teletype)

CRT: für Konsole bzw. Terminal: Leser, Stanzer oder LIST-Einheit (Bildschirm)

PTR: für einen Lochstreifen- oder Lochkartenleser

PTP: für einen Lochstreifen- oder Lochkartenstanzer

LPT: für eine LIST-Einheit (Zeilendrucker)

UC1: für eine vom Benutzer definierte Konsole oder Terminal

UR1: für einen vom Benutzer definierten Leser

UR2: für einen zweiten vom Benutzer definierten Leser

UP1: für einen vom Benutzer definierten Stanzer

UP2: für einen zweiten vom Benutzer definierten Stanzer

UL1: für eine vom Benutzer definierte LIST-Einheit

Wichtig: Die Einheit BAT: wurde nicht aufgeführt, da sie nur die Werte für RDR: und LST: zuordnet.

Anhang G

PIP-Geräte-Codes

NUL: sendet 40 Nullen (ASCII-Code) zur Geräteeinheit (im allgemeinen die Stanzeinheit). Im folgenden Beispiel werden nach der Übertragung des Programms PROG.HEX noch 40 Nullen gesendet (d.h. Leerstreifen erzeugt):

***PUN: = PROG.HEX,NUL:↵**

EOF: sendet ein Dateiendezeichen (EOF = ASCII^Z) zur Geräteeinheit. Bei ASCII-Textdateiübertragungen wird dieses Zeichen automatisch gesendet, so daß dieses nur in einigen Sonderfällen benötigt wird. Im folgenden Beispiel:

***PUN: = NUL:,X.ASM,EOF:,NUL:↵**

werden zunächst zur Stanzeinheit 40 Nullen, dann eine Kopie der Datei ASM, ein nachfolgendes EOF-Zeichen (^Z) und weitere 40 Nullen übertragen.

PRN: entspricht der LST:-Einheit mit der Ausnahme, daß die Tabulatorsteuerungen auf alle 8 Spalten festgesetzt, die Zeilen numeriert und die ausgedruckten Seiten über automatische Formularvorschübe (alle 60 Zeilen, wenn nicht als Option anders vereinbart) gesteuert werden.

***PRN: = BEISPIEL.TXT↵**

INP: ist eine spezielle Gerätebezeichnung, die in das PIP-Programm eingefügt werden kann (patched). Die Einfügung muß dann in Assemblersprache im PIP gesetzt werden. PIP empfängt dann das eingegebene Zeichen über den Aufruf einer bestimmten Speicherposition (103H bei CP/M-80) und legt dieses Zeichen im Speicher ab (auf Adresse 109H bei CP/M-80).

OUT: ist genau wie INP eine spezielle Gerätebezeichnung, die ebenso in das PIP-Programm noch eingefügt werden kann. Bei CP/M-80 muß die dafür notwendige Routine auf Adresse 106H stehen. Das Ausgabezeichen steht im C-Register.

Wichtig: Der für diese „Patches" vorgesehene Speicherbereich ist beim Original-PIP frei. Anstelle der INP- und OUT-Routine steht lediglich ein Rücksprung-Befehl (RET).

Anhang H

PIP-Parameter

A Archivierungs-Option (nicht bei CP/M-80 und CP/M-86). Nur Dateien, die seit der letzten Kopierung (Archivierung) nicht geändert wurden, werden kopiert. Als Kennzeichnung dient bei den Dateien das Archiv-Attribut.

B Übertragung im blockmode. Die PIP-Routine schiebt dabei Daten in einen Puffer, bis ein ASCII-X-off-Zeichen (^S) erkannt wird. Die Größe des Puffers ist abhängig von der Systemkonfiguration (vgl. spezielles Systemhandbuch). Mit diesem Parameter können fortlaufend Daten von einer sequentiellen Geräteeinheit gelesen werden (z.B. Kassettenrecorder).

```
*SERVE.TXT = RDR:[B]↵
```

C Confirm-Option (nur CP/M-Plus). Bei der Übertragung von Dateigruppen muß jede Kopier-Operation bestätigt werden. Beispiel:

```
*B:=a:*.TXT[C]↵
COPYING –
BRIEF.TXT (Y/N)? Y
LISTE.TXT (Y/N)? N
*
```

Mit Y wird das Kopieren bestätigt. N verhindert das Kopieren.

Dn Delete Option. Die PIP-Routine löscht während der Übertragung die über die n-te Spalte am Bildschirm hinausgehenden Zeichen. Hiermit können lange Zeilen für die Übertragung auf Geräteeinheiten mit kurzen Ausgabezeilen ohne Probleme abgeschnitten werden. Mit dem Befehl

```
*PRN: = LANG.TXT[D40]↵
```

können z.B. Kommentare, die immer an bestimmter Spaltenposition beginnen, ausgeblendet werden.

E Alle Kopiervorgänge werden in der Reihenfolge ihrer Bearbeitung am Terminal gemeldet (echoed). Beispiel:

***KOPIE.TXT = QUELL.TXT,S2.TXT,S3.TXT,S4.TXT[E]↵**

Bei einer großen Folge von Übertragungen können diese leicht verfolgt werden.

F Alle Formularvorschubzeichen (FF = form feed) werden ausgefiltert. Genauso können nachträglich noch Formularvorschubzeichen mit dem P-Parameter eingefügt werden. Hiermit ist es dann sehr leicht möglich, eine Bereinigung von Dateien vorzunehmen.

Gn Kopiere Dateien aus einem anderen Benutzerbereich (User-Nummer ist n). Diese Option kann (außer bei CP/M-80) auch hinter der Zieldatei angegeben werden und vereinbart den Benutzerbereich, zu dem kopiert werden soll.

H Hexadezimale Datenübertragung: Die PIP-Routine überprüft die Daten auf zulässiges Intel-Hexadezimal-Format.

I Ignorierung von 00-Sätzen innerhalb der Übertragung von Intel-Hex-Dateien (H-Parameter wird automatisch gesetzt). Auch hier werden HEX-Dateien erwartet.

L Umsetzung aller Großbuchstaben (upper case) in Kleinbuchstaben (lower case).

N Einfügen von Zeilennummern in eine Dateikopie (beginnend bei Zeile 1). Hinter jede Zeilennummer wird ein Semikolon gesetzt. Führende Nullen (z.B. 003) werden weggelassen, es sei denn, es wird der NZ-Parameter gesetzt. Bei NZ werden Nullen nicht weggelassen und hinter die Zahlen Tabulatorzeichen gesetzt. Mit dem T-Parameter können die Tabulatorstände beliebig verändert und damit sehr gut lesbare Ausdrucke generiert werden.

O Übertragung von Objektdateien (keine ASCII-Dateien). Die PIP-Routine ignoriert beim Zusammenfassen von Dateien jeweils das physikalische Dateiende (vgl. Zusammenfassen von Dateien).

Pn PIP generiert nach jeder n-ten Zeile einen Seitenvorschub (intern gesteuerter Seitenvorschub). Wenn n mit 1 oder gar nicht angegeben ist, wird alle 60 Zeichen ein Seitenvorschub erzeugt (vorbestimmt = default). Wenn gleichzeitig auch der F-Parameter gesetzt ist, wird von PIP zunächst das Seitenvorschubzeichen in der Datei gelöscht und danach der ge-

wünschte Seitenvorschub erzeugt. Mit dieser Methode kann sehr leicht eine gewünschte Seitenlänge erreicht werden.

Q string — Die PIP-Routine bricht die Übertragung von einer Geräteeinheit ab, sobald die angegebene Zeichenkette ^Z erkannt wird (eine Zeichenkette oder auch String ist eine Folge einzelner Zeichen; z.B. STRING105%). Der Abschluß der gewählten Zeichenkette wird mit ^Z angegeben. Mit diesem Befehl kann das Kopieren einzelner Dateiabschnitte erreicht werden (vgl. Abschnitt „Kopieren einzelner Dateibereiche").

R — Dateien mit dem System-Attribut werden mitübertragen.

S string ^Z — Die PIP-Routine beginnt mit der Übertragung von einer Geräteeinheit oder einer Datei, sobald die angegebene Zeichenkette erkannt wird. Die Zeichenkette wird mit ^Z abgeschlossen. Mit diesem Parameter kann sehr leicht ein Dateibereich ab einer bestimmten Startposition übertragen werden.

Tn — Die Tabulatorabstände werden während der Übertragung auf n Spalten gesetzt. Innerhalb einer Textdatei können die Tabulatorsteuerzeichen mit ^I generiert werden. Mit dem T-Parameter können so nachträglich andere Tabulierungen erreicht werden.

U — Umsetzung aller Kleinbuchstaben in Großbuchstaben während der Übertragung.

V — Von der PIP-Routine werden alle in die Kopie übertragenen Daten mit der Originaldatei verglichen (verify) und Abweichungen gemeldet.

W — Schreibgeschützte Dateien (R/O-Attribut) auf dem Ziel-Laufwerk werden ungefragt überschrieben.

Z — Das Paritätsbit eingelesener Daten wird auf Null gesetzt.

Folgende Beispiele zeigen die Anwendung der PIP-Parameter im Zusammenhang:

```
*LST: = BEISPIEL.TXT[NT8P60]↵
```

Dieser Ausdruck sendet die Datei BEISPIEL.TXT zur LIST-Einheit (LST:). Sie wird dabei mit Zeilennummern ausgedruckt (N), Steuerzeichen für den Tabulator werden in Schritten von 8 Spalten interpretiert, und alle 60 Zeilen wird ein Seitenvorschub generiert. Wird anstelle der LST:-Einheit die PRN:-Einheit angesprochen, können die angegebenen Parameter entfallen. Das obige Beispiel verkürzt sich dann auf

```
*PRN: = BEISPIEL.TXT↵
```

Es ist möglich, die vorgegebenen Werte der PRN-Einheit mit neuen Werten selektiv zu überschreiben:

∗PRN: = BEISPIEL.TXT[P59]↵

Die Datei BEISPIEL.TXT wird mit allen vorgegebenen (default) Parametern (z.B. NT8), ausgenommen der Seitenvorschub (jetzt 59 Zeilen), zur PRN:-Einheit gesendet.

Mit dem folgenden Ausdruck

∗LST: = PROG.ASM[NT8U]↵

wird das Programm PROG.ASM mit Zeilennummern (N), Tabulatorschritten in jeweils 8 Spalten (T8) und in Großbuchstaben (U) zur Listeinheit übertragen.

Anhang I

CP/M- und MP/M-Befehle (Zusammenfassung)

Befehl	CP/M-80	CP/M-Plus	MP/M-80	CP/M-86	MP/M-86	CCP/M
ABORT			X		X	X
ASM	X		X			
ASM86				X	X	X
ATTACH			X		X	
CONSOLE			X		X	
DATE		X				
DDT	X		X			
DDT86				X	X	X
DIR	X	X	X	X	X	X
DSKRESET			X		X	X
DUMP	X	X	X			
ED	X	X	X	X	X	X
ERA	X	X	X	X	X	X
ERAQ			X		X	X
GENCMD				X	X	X
GENHEX			X			
HELP		X				X
INITDIR		X				
LINK		X	X			
LOAD	X		X			
MOVCPM	X					
MPMLDR			X		X	
MPMSTAT			X		X	
PIP	X	X	X	X	X	X

Befehl	**CP/M-80**	**CP/M-Plus**	**MP/M-80**	**CP/M-86**	**MP/M-86**	**CCP/M**
PRINTER			X		X	X
PRLCOM			X			
REN	X	X	X	X	X	X
RMAC		X	X			
SAVE	X	X				
SCHED			X			
SDIR			X		X	X
SET		X	X		X	X
SHOW		X	X		X	X
SID		X	X			
SPOOL			X		X	
STAT	X		X	X	X	X
STOPSPLR			X		X	
SUBMIT	X	X	X	X	X	X
SYSGEN	X					
TOD			X		X	X
TYPE	X	X	X	X	X	X
USER	X	X	X	X	X	X
XSUB	X					

Anhang **J**

Steuerzeichen bei der Befehlseingabe

Auslösende Taste	Operation
DELETE (RUBOUT)	Löschen des letzten Zeichens mit Echo
BACKSPACE (BS)	Löschen des letzten Zeichens mit Cursor zurück
CTRL-U	Löschen der gesamten Zeile, Ausgabe von # auf der nächsten Zeile
CTRL-X	Löschen der gesamten Zeile und Rücksetzen des Cursors auf Zeilenanfang
CTRL-R	Wiederholung der Eingabezeile auf dem Terminal
RETURN oder LINEFEED	Abschluß der Eingabezeile
CTRL-E	Eine neue Zeile auf dem Terminal wird begonnen. Kein Einfluß auf das Kommando
CTRL-C	Bei Betätigung am Zeilenanfang: Warmstart für CP/M
CTRL-D	nur MP/M: DETACH
CTRL-P	Ein- und Ausschalten des Druckerechos. Druckerecho bedeutet, daß alle zum Terminal gesandten Zeichen auch zum Drucker gesendet werden.
CTRL-S	Stoppt die Ausgabe von Texten auf das Terminal
CTRL-Q	Startet die Ausgabe auf das Terminal nach ^S

Anhang K
Dateikennungen

Dateityp	Zuordnung	Beispiel
COM	Kommando-Datei für die 80er-Prozessoren	PIP.COM
CMD	Kommando-Datei für die 86er-Prozessoren	ED.CMD
68K	Kommando-Datei für 68000-Prozessor	DDT.68K
PRL	Kommando-Datei für MP/M-80	SDIR.PRL
ASM	Assembler-Quellprogramm für 80er-Prozessoren	EXAMP.ASM
A68	Assembler-Quellprogramm für 86er-Prozessoren	PATCH.A68
PRN	vom Assembler erzeugtes Listing	EXAMP.PRN
HEX	Programmcode in [HEX]-Format	EXAMP.HEX
H86	Programmcode in erweitertem [HEX]-Format	PATCH.H86
BAS	BASIC-Quellprogramm	GAME.BAS
INT	Zwischencode z.B. für Interpreter	GAME.INT
BAK	Sicherungsdatei (backup) des Editors	EXAMP.BAK
$$$	Zwischendatei nur während Programmlauf	LETTER.$$$
SUB	Textdatei für autom. Programmablauf (batch)	COMPIL.SUB
SPR	Systemfile für MP/M und CP/M-Plus	XDOS.SPR
RSP	„Residenter System Prozess“ (MP/M)	SPOOL.RSP
BRS	wie RSP, jedoch für Banking	SPOOL.BRS
REL	„Relocatible“ (verschiebbarer Objektcode)	BDOS3.REL
IRL	„Indexed Relocatable“ (spezieller REL-Code)	BASLIB.IRL

Anhang L

Prüfliste für Zubehör

- ○ Leere Disketten
- ○ Systemdiskette
- ○ Disketten mit den Anwendungsprogrammen
- ○ Typenräder
- ○ Farbbänder
- ○ Endlospapier
- ○ Steckdosenleiste mit Netzschalter
- ○ Computerhandbuch
- ○ CRT-Terminalhandbuch
- ○ Druckerhandbuch
- ○ CP/M-Dokumentation
- ○ Dokumentation der Anwendungsprogramme

Anhang N

Fehler-Prüfliste

Totalausfall

- ○ Überprüfen der mechanischen Verbindungen:
  - Stromversorgungsleitungen
  - Verbindungskabel
  - Schalterstellungen
  - Sicherungen

Druckerausfall

- ○ Testversuch im „local"-Betrieb
- ○ Eingabe von CTRL-P von der Konsole aus
- ○ Überprüfung aller Schaltereinstellungen
- ○ Korrektes Einlegen des Druckpapiers
- ○ Überprüfen der Sicherungen

Drucker stoppt nicht

- ○ Eingabe von CTRL-P
- ○ Eingabe von CTRL-C
- ○ Ausschalten des Druckers

Systemausfall

- ○ System-Wiederstart (Warmstart; CTRL-C)
- ○ Abbruch und erneuter vollständiger Systemstart

Laufwerkskontrollampe erlischt nicht

- ○ keine Diskette im Laufwerk
- ○ Einlegen einer anderen Systemdiskette und neuer Start

Nicht identifizierbares Systemverhalten

- Verdacht auf Bedienungsfehler. Erneuter Versuch. Überprüfung der Systemdiskette und korrekter Schalterstellungen am Drucker.
- Verdacht auf zerstörte Systemdiskette. Ersatz gegen neue Kopie.
- Verdacht auf zerstörtes Anwendungsprogramm. Ersatz gegen neue Kopie.
- Ausschalten der gesamten Anlage und erneuter Versuch.
- Verdacht auf Hardwarefehler.

Anhang O

Begriffe und Abkürzungen

Fachbegriffe – insbesondere englische – und eine Vielzahl von Abkürzungen erschweren dem Anfänger oft den Einstieg in die Praxis der Personal Computer. Andererseits läßt sich im Sinne einer kurzen und klaren Darstellung dieser Fachjargon nur schwer vermeiden. Die in der folgenden Tabelle zusammengestellten Fachbegriffe und Abkürzungen sollen Ihnen das Verständnis des CP/M-Handbuches erleichtern. Weitere Informationen finden Sie im Mikrocomputer Lexikon, SYBEX-Verlag.

Englisch	**Deutsch**
backspace	Rücktaste (Eingabetastatur)
(memory) bank	Speicherbank
baud	Baud (Übertragungsrate) 1 Baud entspricht 1 Bit / s
bit	Bit, kleinste Informationseinheit mit den Werten „0“ und „1“
bootstrap	Startroutine
breakpoint	Haltepunkt
byte	Byte (8 Bit)
call	Aufruf (einer Funktion oder eines Programms)
carriage return	Wagenrücklauf-Taste (kurz: return)
cold start	Kaltstart (z.B. von CP/M nach Einschalten des elektrischen Netzes
concatenation	Verkettung (z.B. von Dateien)
debugger	Testhilfe
default	Vorgabe-Vereinbarung
delete	Lösch-Taste (auch rubout)
directory	Inhaltsverzeichnis
disk	Magnetplatte
diskette	Diskette, Floppy, flexible Magnetplatte
dispatcher	Arbeitsverteiler
drive	(Platten-) Laufwerk
enter	Andere Bezeichnung für carriage return
file	Datei

floppy	Flexible Magnetplatte
form feed	Seitenvorschub (beim Drucker)
hardware	Hardware (z.B. Personal Computer, Laufwerke, Drucker)
joystick	Steuerknüppel
label	Marke, Aufkleber
light pen	Lichtgriffel
line feed	Zeilenvorschub (auf dem Drucker)
logged disk	Angewähltes Laufwerk
multi-tasking	Multi-Tasking (parallele Bearbeitung mehrerer Aufgaben)
password	Paßwort
prompt	Quittung
queue	Warteschlange
read only	Nur-Lese-Modus
read/write	Schreib-/Lese-Modus
record	Datensatz
return	Siehe carriage return
rubout	Lösch-Taste (wie delete-Taste)
single density	Einfache Schreibdichte (Floppy)
single sided	Einseitig (Floppy)
skew factor	Standardabweichung
soft sectored	Soft-Sektoriert (Floppy)
software	Software
task	Task (Aufgabe)
terminal	Terminal (Bildschirm und Tastatur)
trace	Protokollieren
trap	Falle
track	Spur (auf der Platte)
utility	Dienstprogramm
warm start	Warmstart
wildcard	Joker, Platzhalter

Abkürzung	**Langform**
ASCII	American Standard Code for Information Exchange
ASM	Assembler
BDOS	Basic Disk Operating System
BIOS	Basic Input/Output System
BS	Backspace (Rücktaste)
CCP	Console Command Processor (CP/M)
CCP/M	Concurrent CP/M
CLI	Command Line Interpreter (MP/M)

CP	Character Pointer
CP/M	Control Program for Microprocessors
CP/M Plus	CP/M Version 3.0
CR	Carriage Return
CRT	Cathode Ray Tube (Bildschirm)
CTRL	Control-Taste
DDT	Dynamic Debugging Tool
DEL	Delete-Taste
DIR	DIRectory-Befehl
DOS	Disk Operating System
DUMP	DUMP-Befehl
ED	EDitor-Befehl
EOF	End Of File (^Z)
ERA	ERAse-Befehl
ESC	ESCape-Taste
FCB	File Control Block
FF	Form Feed (Seitenvorschub)
IC	Integrated Circuit (Integrierter Schaltkreis)
KB	kByte, Kilo-Byte, 1 KB = 1024 Byte
LF	Line Feed (Zeilenvorschub)
MB	MByte, Mega-Byte, 1 MB = 1024 KB
MP/M	Multi-Programming Monitor Control Program for Microprocessors
PD	Process Descriptor (MP/M)
PIP	Peripheral Interchange Program
R/O	Read Only
R/W	Read / Write
RAM	Random Access Memory (Schreib-/Lesespeicher)
REN	REName-Befehl
ROM	Read-Only Memory (Nur-Lesespeicher)
SID	Symbolic Instruction Debugger (Digital Research)
TAB	TABulator-Taste
TMP	Terminal Message Processor (MP/M)
TOD	Time-Of-Day-Befehl
TPA	Transient Program Area (Arbeitsspeicher)
XDOS	EXtended Disk Operating System
XFCB	EXtended File Control Block
XIOS	EXtended Input/Output System
XSUB	EXtended Submit-Befehl
ZSID	Z80 Symbolic Instruction Debugger

Anhang **P**

Einige Adressen

In einigen Kapiteln wurden Software-Produkte aufgeführt. Unter Umständen ist es nicht möglich, das eine oder andere Programm zu beschaffen, oder eine spezielle Frage kann nicht beantwortet werden. Für diesen Fall ist es interessant, die Adresse des betreffenden Herstellers zu besitzen.

Internationale CP/M-Benutzergruppe (CPMUG Library):

CPMUG CP/M Users Group
Postfach 1213
7590 Achern 2

Softwarehaus, Entwicklung und Vertrieb der CP/M-Familie:

Digital Research
Hansastraße 15
8000 München

Softwarehaus (ASCOM):

Dynamic Microprocessor Ass.
545 Fifth Avenue, Room 602
New York, NY 10 017
USA

Europäische CP/M-Benutzergruppe:

European CP/M Users Club
Hans-Toma-Straße 10
7515 Linkenheim

Softwarehaus (DI Disk Inspector):

Overbeek Enterprises
P.O. Box 726
Elgin, IL 60 120
USA

Softwarehaus (ZAP):

Phase Four Software Inc.
55 Ridgewood Terrace
Chappaqua, NY 10 514
USA

Softwarehaus (Diagnostic, Disk Doctor):

Supersoft Inc.
P.O. Box 1628
Champaign, IL 61 820

Stichwortverzeichnis

Die SYBEX Bibliothek

Einführende Literatur

MEIN ERSTER COMPUTER
von Rodnay Zaks – Der unentbehrliche Wegweiser für jeden, der den Kauf oder den Gebrauch eines Mikrocomputers erwägt, das Standardwerk in 3., überarbeiteter Ausgabe. 304 Seiten, 150 Abbildungen, zahlreiche Illustrationen, Best.-Nr.: **3040** (1984)

VORSICHT! Computer brauchen Pflege
von Rodnay Zaks – das Buch, das Ihnen die Handhabung eines Computersystems erklärt – vor allem, was Sie damit nicht machen sollten. Allgemeingültige Regeln für die pflegliche Behandlung Ihres Systems. 240 Seiten, 96 Abbildungen, Best.-Nr.: **3013** (1983)

MIT DEM COMPUTER UNTERWEGS
von W. Höfs – für alle, die einen netzunabhängigen Rechner benötigen; alles über Handheld-Computer. 144 S., 27 Abb., Best.-Nr. **3067** (1984)

CHIP UND SYSTEM: Einführung in die Mikroprozessoren-Technik
von Rodnay Zaks – eine sehr gut lesbare Einführung in die faszinierende Welt der Computer, vom Microprozessor bis hin zum vollständigen System. 576 Seiten, 325 Abbildungen, Best.-Nr.: **3017** (1984)

EINFÜHRUNG IN DIE TEXTVERARBEITUNG
von Hal Glatzer – beschreibt, woraus eine Textverarbeitungsanlage besteht, wie man sie nutzen kann und wozu sie fähig ist. Beispiele verschiedener Anwendungen und Kriterien für den Kauf eines Systems. 248 Seiten, 67 Abbildungen, Best.-Nr. **3018** (1983)

SYBEX MIKROCOMPUTER LEXIKON
– die schnelle Informationsbörse! Über 1500 Definitionen, Kurzformeln, Begriffsschema der Mikroprozessor-Technik, englisch/deutsches und französisch/deutsches Wörterbuch, Bezugsquellen. 192 Seiten, Format 12,5 x 18 cm, Best.-Nr.: **3035** (1984)

COMPUTER TOTAL VERRÜCKT
von Daniel Le Noury – mit diesem Buch kommen Sie wieder zur Besinnung, nachdem Sie sich halbtot gelacht haben. Ca. 100 Cartoons rund um den Computer. 96 Seiten, Best.-Nr. **3042** (1984)

BASIC

BASIC COMPUTER SPIELE/Band 1
herausgegeben von David H. Ahl – die besten Mikrocomputerspiele aus der Zeitschrift „Creative Computing" in deutscher Fassung mit Probelauf und Programmlisting. 208 Seiten, 56 Abbildungen, Best.-Nr. **3009**

BASIC COMPUTER SPIELE/Band 2
herausgegeben von David H. Ahl – 84 weitere Mikrocomputerspiele aus „Creative Computing". Alle in Microsoft-BASIC geschrieben mit Listing und Probelauf. 224 Seiten, 61 Abbildungen, Best.-Nr.: **3010**

MEIN ERSTES BASIC PROGRAMM
von Rodnay Zaks — das Buch für Einsteiger! Viele farbige Illustrationen und leichtverständliche Diagramme bringen Spaß am Lernen. In wenigen Stunden schreiben Sie Ihr erstes nützliches Programm. 208 Seiten, illustriert, Best.-Nr.: **3033** (1983)

BASIC PROGRAMME — MATHEMATIK, STATISTIK, INFORMATIK
von Alan Miller — eine Bibliothek von Programmen zu den wichtigsten Problemlösungen mit numerischen Verfahren, alle in BASIC geschrieben, mit Musterlauf und Programmlisting. 352 Seiten, 147 Abbildungen, Best.-Nr.: **3015** (1983)

PLANEN UND ENTSCHEIDEN MIT BASIC
von X. T. Bui — eine Sammlung von interaktiven, kommerziell-orientierten BASIC-Programmen für Management- und Planungsentscheidungen. 200 Seiten, 53 Abbildungen, Best.-Nr.: **3025** (1983)

BASIC FÜR DEN KAUFMANN
von D. Hergert — das BASIC-Buch für Studenten und Praktiker im kaufmännischen Bereich. Enthält Anwendungsbeispiele für Verkaufs- und Finanzberichte, Grafiken, Abschreibungen u.v.m. 208 Seiten, 85 Abbildungen, Best.-Nr.: **3026** (1983)

PLANEN, KALKULIEREN, KONTROLLIEREN MIT BASIC-TASCHENRECHNERN
von P. Ickenroth — präsentiert eine Reihe von direkt anwendbaren BASIC-Programmen für zahlreiche kaufmännische Berechnungen mit Ihrem BASIC-Taschenrechner. 144 Seiten, 48 Abbildungen, Best.-Nr.: **3032** (1983)

Pascal

EINFÜHRUNG IN PASCAL UND UCSD/PASCAL
von Rodnay Zaks — das Buch für jeden, der die Programmiersprache PASCAL lernen möchte. Vorkenntnisse in Computerprogrammierung werden nicht vorausgesetzt. Eine schrittweise Einführung mit vielen Übungen und Beispielen. 535 Seiten, 130 Abbildungen, Best.-Nr.: **3004** (1982)

DAS PASCAL HANDBUCH
von Jacques Tiberghien — ein Wörterbuch mit jeder Pascal-Anweisung und jedem Symbol, reservierten Wort, Bezeichner und Operator, für beinahe alle bekannten Pascal-Versionen. 480 Seiten, 270 Abbildungen, Format 23 x 18 cm, Best.-Nr.: **3005** (1982)

PASCAL PROGRAMME — MATHEMATIK, STATISTIK, INFORMATIK
von Alan Miller — eine Sammlung von 60 der wichtigsten wissenschaftlichen Algorithmen samt Programmauflistung und Musterdurchlauf. Ein wichtiges Hilfsmittel für Pascal-Benutzer mit technischen Anwendungen. 398 Seiten, 120 Abbildungen, Format 23 x 18 cm, Best.-Nr.: **3007** (1982)

50 PASCAL-PROGRAMME
von B. Hunter — eine kommentierte Sammlung nützlicher Programme für Anwendungen im Geschäft und Privatbereich, für mathematische Anwendungen oder Spiele. Ca. 340 S., ca. 48 Abb., Best.-Nr. **3065** (erscheint 1985)

GRUNDKURS IN PASCAL Bd. 1
von K.-H. Rollke — der sichere Einstieg in Pascal, speziell für Schule und Fortbildung (Reihe SYBEX Informatik). 224 Seiten, mit Abb., Format 17,5x25 cm, Best.-Nr. **3046** (1984), Lehrerbegleitheft Best.-Nr. **3059**

GRUNDKURS IN PASCAL BAND 2
von K.H. Rollke – Mit diesem Buch wird der Pascal-Grundkurs aus der Reihe SYBEX Informatik abgerundet. Für Lehrer, Schüler, Teilnehmer an Pascal-Kursen, Studenten und Autodidakten. 224 Seiten, mit Abb., Best.-Nr. **3061** (erscheint 1985), Lehrerbegleitheft Best.-Nr. **3090**

C

ERFOLGREICH PROGRAMMIEREN MIT C
von J. A. Illik – ein unentbehrliches Handbuch für jeden, der mit der universellen Sprache C erfolgreich programmieren will. Aussagekräftige Beispiele, auf verschiedenen Mini- und Mikrocomputern getestet. 408 Seiten, Best.-Nr.: **3055** (1984)

Assembler

PROGRAMMIERUNG DES Z80
von Rodnay Zaks – ein kompletter Lehrgang in der Programmierung des Z80 Mikroprozessors und eine gründliche Einführung in die Maschinensprache. 608 Seiten, 176 Abbildungen, Format DIN A5, Best.-Nr.: **3006** (1982)

Z80 ANWENDUNGEN
von J. W. Coffron – vermittelt alle nötigen Anweisungen, um Peripherie-Bausteine mit dem Z80 zu steuern und individuelle Hardware-Lösungen zu realisieren. 296 Seiten, 204 Abbildungen, Best.-Nr.: **3037** (1984)

FORTGESCHRITTENE 6502-PROGRAMMIERUNG
von Rodnay Zaks – hilft Ihnen, schwierige Probleme mit dem 6502 zu lösen, stellt Ihnen Maschinenroutinen zum Arbeiten mit einem Hobbyboard vor. 288 Seiten, 140 Abbildungen, Best.-Nr.: **3047** (1984)

6502 ANWENDUNGEN
von Rodnay Zaks – das Eingabe-/Ausgabe-Buch für Ihren 6502-Microprozessor. Stellt die meistgenutzten Programme und die dafür notwendigen Hardware-Komponenten vor. 288 Seiten, 213 Abbildungen, Best.-Nr.: **3014** (1983)

PROGRAMMIERUNG DES 6809
von R. Zaks und W. Labiak – eine vollständige Einführung in die Assemblerprogrammierung mit dem 6809, für alle, die mit DRAGON 32, Tandy Colorcomputer oder einem anderen 6809-System arbeiten. 400 Seiten, 150 Abbildungen, Best.-Nr.: **3049** (1984)

PROGRAMMIERUNG DES 8086/8088
von J. W. Coffron – lehrt Sie Programmierung, Kontrolle und Anwendung dieses 16-Bit-Mikroprozessors; vermittelt Ihnen das notwendige Wissen zu optimaler Nutzung Ihrer Maschine, von der internen Architektur bis hin zu fortgeschrittenen Adressierungstechniken. 312 Seiten, 107 Abbildungen, Best.-Nr.: **3050** (1984)

Spezielle Geräte

Apple

APPLE II LEICHT GEMACHT
von J. Kascmer – macht Sie schnell mit Tastatur, Bildschirm und Diskettenlaufwerken vertraut. Sie lernen, wie leicht es ist, Ihr eigenes BASIC-Programm zu schreiben. 192 Seiten, mit 43 Abbildungen, Best.-Nr.: **3031** (1984)

BASIC ÜBUNGEN FÜR DEN APPLE
von J.-P. Lamoitier – das Buch für APPLE-Nutzer, die einen schnellen Zugang zur Programmierung in BASIC suchen. Abgestufte Übungen mit zunehmendem Schwierigkeitsgrad. 256 Seiten, 190 Abbildungen, Best.-Nr.: **3016** (1983)

APPLE II BASIC HANDBUCH
von D. Hergert – ein handliches Nachschlagewerk, das neben Ihren Apple II, II+ oder IIe stehen sollte. Dank vieler Tips und Vorschläge eine wesentliche Erleichterung fürs Programmieren. 304 Seiten, 116 Abbildungen, Best.-Nr. **3036** (1984)

PROGRAMME FÜR MEINEN APPLE II
von S. R. Trost – enthält eine Reihe von lauffähigen Programmen samt Listing und Beispiellauf. Hilft Ihnen, viele neue Anwendungen für Ihren APPLE II zu entdecken und erfolgreich einzusetzen. 192 Seiten, 158 Abbildungen, Best.-Nr.: **3029** (1983)

APPLE II/IIe ASSEMBLER KURS
Reihe MISTER MICRO – Assembler-Programmierung auf dem Apple leicht gemacht. Das Buch vermittelt alle Instruktionen für den 6502-Prozessor. Der Assembler kann jederzeit für eigene Programme eingesetzt werden. 240 Seiten, Buch und Diskette, Best.-Nr. **3408** (1984)

ARBEITEN MIT DEM MACINTOSH
von N. Hesselmann – alles über den leistungsfähigen Apple-Rechner mit einer Erläuterung wichtiger kommerzieller Software-Pakete und deren Einsatz, Anleitung zur Programmierung in Microsoft-BASIC. Viele konkrete Anwendungs-Beispiele. 416 Seiten, 320 Abb., Best.-Nr. **3080** (1984)

Atari

ATARI BASIC HANDBUCH
von J. Reschke – das vollständige ABC der BASIC-Programmierung für den Atari mit Erläuterung durch viele praktische Beispiele. 208 Seiten, mit Abb., Best.-Nr. **3083** (1984)

ATARI PROGRAMM-SAMMLUNG
von S. R. Trost – sollte neben keinem Atari-Computer fehlen. Es bietet einen Satz ausgetesteter Programme für eine Fülle von Anwendungen. 190 S., 150 Abb., Best.-Nr. **3068** (1984)

Commodore

MEIN ERSTES COMMODORE 64-PROGRAMM
von R. Zaks – sollte Ihr erstes Buch zum Commodore 64 sein. Viel Spaß am Lernen durch farbige Illustrationen und leichtverständliche Diagramme, Programmieren mit sofortigen Resultaten. 208 Seiten, illustriert, Best.-Nr. **3062** (1984)

MEIN ZWEITES COMMODORE 64 PROGRAMM
von Gary Lippman – für alle, die bereits ein Grundwissen in BASIC haben und mit ihrem C 64 den nächsten Schritt machen wollen – und das mit viel Spaß. 240 Seiten, zahlr. witzige Illustr., Best.-Nr. **3086** (erscheint 1985)

COMMODORE 64 – LEICHT GEMACHT
von J. Kascmer – führt Sie schnell in die Bedienung von Tastatur, Bildschirm und Diskettenlaufwerken ein, macht Sie zum BASIC-Programmierer Ihres C64! 176 Seiten, 36 Abbildungen, Best.-Nr.: **3038** (1984)

COMMODORE 64 BASIC HANDBUCH
von D. Hergert – zeigt Ihnen alle Anwendungsmöglichkeiten Ihres C64 und beschreibt das vollständige BASIC-Vokabular anhand von praktischen Beispielen. 208 Seiten, 92 Abbildungen, Best.-Nr.: **3048** (1984)

FARBSPIELE MIT DEM COMMODORE 64
von W. Black und M. Richter – 20 herrliche Farbspiele für Ihren C64, mit Beschreibung, Programmlisten und Bildschirm-Darstellungen. Für mehr Freizeit-Spaß mit Ihrem Commodore! 176 Seiten, 58 Abbildungen, Best.-Nr.: **3044** (1984)

COMMODORE 64 – GRAFIK + DESIGN
von Ch. Platt – Eine Schritt-für-Schritt-Einführung in die Grafik-Programmierung Ihres C 64. Tips, die Sie in keinem Handbuch finden. 280 S., 150 Abb., teils vierfarbig. Best.-Nr. **3073** (1984)

COMMODORE 64 PROGRAMMSAMMLUNG
von S. R. Trost – mehr als 70 getestete Anwenderprogramme, die direkt eingegeben werden können. Erläuterungen gewährleisten eine optimale Nutzung. 192 Seiten, 160 Abbildungen, Best.-Nr.: **3051** (1983)

SPASS AN MATHE MIT DEM COMMODORE 64
von H. Danielsson – zeigt Ihnen mit vielen Beispielen, wie der C 64 für schulische oder private Berechnungen genutzt werden kann. 280 Seiten mit Abb., Best.-Nr. **3072** (erscheint 1985)

COMMODORE 64 BASIC ABENTEUER
Der fremde Planet
Reihe MISTER MICRO – Für Kinder und Jugendliche. Sie unternehmen eine Reise ins All und lernen dabei auf unterhaltsame Weise, den C64 in BASIC zu programmieren. 136 Seiten, Buch und Kassette, Best.-Nr. **3404**, Buch und Diskette, Best.-Nr. **3405** (1984)

COMMODORE 64 BASIC-KURS MIT HONEY-AID
Reihe MISTER MICRO – BASIC auf dem C64 durch Praxis lernen; mit dem integrierten Lernpaket (Buch + Software). Außer vielen Übungsprogrammen: Honey-Aid – eine universell einsetzbare BASIC-Erweiterung mit 28 zusätzlichen Befehlen. 352 Seiten, Buch und Kassette, Best.-Nr. **3400**, Buch und Diskette, Best.-Nr. **3401** (1984)

COMMODORE 64 ASSEMBLER-KURS
Reihe MISTER MICRO – zeigt in Theorie und Praxis, wie Sie den 6510-Prozessor Ihres C64 programmieren. Der mitgelieferte Assembler ist universell einsetzbar. 296 Seiten, Buch und Kassette, Best.-Nr. **3402**, Buch und Diskette, Best.-Nr. **3403** (1984)

VC 20 BASIC ABENTEUER
Der fremde Planet
Reihe MISTER MICRO – Der spannendste Weg, BASIC auf dem VC20 zu lernen – mit dem integrierten Lernpaket von Buch und Software. 144 Seiten, Buch und Kassette, Best.-Nr. **3407** (1984)

VC 20 ASSEMBLER-KURS
Reihe MISTER MICRO – Das Lernpaket mit Buch und Software zeigt Ihnen schnell, auf welche Befehle der 6502-Prozessor Ihres VC20 hört. Der Datenträger enthält einen voll funktionsfähigen Assembler. 272 Seiten, Buch und Kassette, Best.-Nr. **3406** (1984)

IBM PC

ARBEITEN MIT DEM IBM PC
von J. Lasselle und C. Ramsay – zeigt Ihnen Schritt für Schritt, wie Sie den IBM PC ohne Vorkenntnisse einsetzen, die speziellen Eigenschaften dieses Computers für Druck, Grafik und Kommunikation nutzen können. 160 Seiten, 25 Abbildungen, Best.-Nr.: **3056** (1984)

BASIC ÜBUNGEN FÜR DEN IBM PERSONAL COMPUTER
von J.-P. Lamoitier – vermittelt Ihnen BASIC durch praktische und umfassende Übungen anhand von realistischen Programmen: Datenverarbeitung, Statistik, kommerzielle Programme, Spiele u.v.m. 256 Seiten, 192 Abbildungen, Best.-Nr.: **3023** (1983)

PROGRAMMSAMMLUNG ZUM IBM PERSONAL COMPUTER
von S. R. Trost – mehr als 65 getestete, direkt einzugebende Anwenderprogramme, die eine weite Palette von kaufmännischen, persönlichen und schulischen Anwendungen abdecken. 192 Seiten, 158 Abbildungen, Best.-Nr.: **3024** (1983)

IBM PC – GRAFIK FÜR DEN KAUFMANN
von N. Ford – komplette Beispielprogramme zeigen Ihnen, wie Sie Ihre eigenen Programme für kommerzielle Grafiken auf dem IBM PC erstellen. 280 Seiten, 74 Abb., Best.-Nr. **3076** (erscheint 1985)

Sinclair

SINCLAIR ZX SPECTRUM BASIC HANDBUCH
von D. Hergert – eine wichtige Hilfe für jeden SPECTRUM-Anwender. Gibt eine Übersicht aller BASIC-Begriffe, die auf diesem Rechner verwendet werden können, und erläutert sie ausführlich anhand von Beispielen. 288 Seiten, 188 Abbildungen, Best.-Nr.: **3027** (1983)

SINCLAIR ZX SPECTRUM Programme zum Lernen und Spielen
von T. Hartnell – ein Buch zur praktischen Anwendung. Grundzüge des Programmierens aus dem kaufmännischen Bereich sowie Spiele, Lehr- und Lernprogramme in BASIC. 232 Seiten, 140 Abbildungen, Best.-Nr. **3022** (1983)

SPECTRUM BASIC ABENTEUER
Der fremde Planet
Reihe MISTER MICRO – Kinder und Jugendliche lernen mit dem kompletten Lernpaket auf spielerische Weise, den Spectrum in BASIC zu programmieren. 128 Seiten, Buch und Kassette, Best.-Nr. **3410** (1984)

SPECTRUM BASIC KURS
Reihe MISTER MICRO – BASIC auf dem Spectrum schnell gelernt mit dem integrierten Lernpaket, das die Programmierung in Theorie und Praxis vermittelt. 312 Seiten, Buch und Kassette, Best.-Nr. **3409** (1984)

MEIN SINCLAIR ZX81
von D. Hergert – eine gut lesbare Einführung in diesen Einplatinencomputer und dessen Programmierung in BASIC. 176 Seiten, 47 Abbildungen, Best.-Nr.: **3021** (1983)

SINCLAIR ZX81 BASIC HANDBUCH
von D. Hergert – vermittelt Ihnen das vollständige BASIC-Vokabular anhand von praktischen Beispielen, macht Sie zum Programmierer Ihres ZX81. 181 Seiten, 120 Abbildungen, Best.-Nr.: **3028** (1983)

SINCLAIR QL BASIC HANDBUCH
von Helmar W. Reuter – anhand vieler praktischer Beispiele lernen Sie das Super-BASIC-Vokabular kennen, die Sprache, die Ihr Sinclair QL versteht. Ca. 300 Seiten, mit Abb., Best.-Nr. **3092** (erscheint 1985)

andere Geräte

MEIN COLOUR GENIE
von Ralf Marquis – zeigt Ihnen, wie man mit einfachen Mitteln eindrucksvolle Programme für den Colour-Genie erstellen kann; viele praktische Beispiele. 160 Seiten, 78 Abb., Best.-Nr. **3063** (1984)

MEIN DRAGON 32
von N. Hesselmann – entwickelt Ihre Fähigkeiten in der Nutzung, Programmierung und erweiterten Anwendung Ihres Rechners anhand von vielen Beispielprogrammen. 256 Seiten, 41 Abbildungen, Best.-Nr.: **3041** (1984)

SCHNEIDER CPC 464: MEIN ERSTES BASIC PROGRAMM
von Rodnay Zaks – zahlreiche farbige Illustrationen und viele Diagramme helfen Ihnen, auf spielerische Weise in BASIC zu programmieren; ohne Vorkenntnisse nutzbar. Ca. 208 Seiten, zahl. farb. Abb., Best.-Nr. **3096** (erscheint 1985)

PLANEN + ENTSCHEIDEN MIT DEM SHARP PC-1500
von X. T. Bui/H. Klein – eine Sammlung interaktiver, kommerziell orientierter BASIC-Programme für Analysen, Planung und Prognosen. Auch für Tandy PC-2. 224 Seiten, 50 Abb., Best.-Nr. **3069** (1984)

SVI PROGRAMM-SAMMLUNG
von S. R. Trost – Knapp 70 ausgetestete Anwenderprograme, u. a. für kommerzielle Berechnungen, Dateiverwaltung und mathematische Übungen; ohne Vorkenntnisse nutzbar. 192 Seiten, 160 Abb., Best.-Nr. **3074** (1984)

SPIELEN, LERNEN, ARBEITEN mit dem TI99/4A
von K.-J. Schmidt und G.-P. Raabe – eine eingehende Einführung in die Bedienung und Programmierung des TI99/4A. Mit den vielen Beispielprogrammen holen Sie das Beste aus Ihrem Computer heraus. 192 Seiten, 41 Abbildungen, Best.-Nr.: **3039** (1984)

Systemsoftware

IBM PC-DOS HANDBUCH
von R. A. King – eine umfassende Einführung in das Disketten-Betriebssystem Ihres IBM PC, seine grundsätzlichen Möglichkeiten und Funktionen sowie auch fortgeschrittene Funktionen (einschließlich der Version 2.0). 320 Seiten, 50 Abbildungen, Best.-Nr.: **3034** (1984)

MS-DOS HANDBUCH
von Richard Allen King – Das unentbehrliche Standardwerk für Anwender des Betriebssystems MS-DOS, abgerundet durch Tabellen, Zeichnungen und viele praktische Beispiele, hilft Ihnen, das Beste aus Ihrem System herauszuholen. Ca. 350 Seiten, 37 Abb., Best.-Nr. **3097** (erscheint 1985)

CP/M-HANDBUCH
von Rodnay Zaks – das Standardwerk über CP/M, das meistgebrauchte Betriebssystem für Mikrocomputer. Für Anfänger eine verständliche Einführung, für Fortgeschrittene ein umfassendes Nachschlagewerk über die CP/M-Versionen 2.2, 3.0 und CCP/M-86 sowie MP/M., 2. überarbeitete Ausgabe. 356 Seiten, 56 Abbildungen, Best.-Nr.: **3053** (1984)

PROGRAMMIEREN MIT CP/M
von A. R. Miller – vermittelt die Feinheiten von CP/M und hilft, die Möglichkeiten dieses populären Betriebssystems zu erweitern. 424 Seiten, ca. 100 Abb., Best.-Nr. **3077** (1985)

UNIX-HANDBUCH
von R. Detering – eine systematische Einführung in UNIX, das kommende Betriebssystem für 16-bit-Rechner. Lernen Sie, Ihren Prozessor optimal einzusetzen! 392 Seiten, 37 Abbildungen, Best.-Nr.: **3054** (1984)

MIKROPROZESSOR INTERFACE TECHNIKEN (3. überarbeitete Ausgabe)
von Rodnay Zaks/Austin Lesea – Hardware- und Software-Verbindungstechniken samt Digital/Analog-Wandler, Peripheriegeräte, Standard-Busse und Fehlersuchtechniken. 432 Seiten, 400 Abbildungen, Format DIN A5, Best.-Nr.: **3012** (1982)

V24/RS-232 KOMMUNIKATION
von J. Campbell – zeigt Ihnen, wie Sie mit den Schnittstellen V24 und RS-232 notwendiges Zubehör an Ihren Rechner anschließen; mit praktischen Fallbeispielen. 224 Seiten, 97 Abb., Best.-Nr. **3075** (1984)

Anwendungssoftware

EINFÜHRUNG IN WORDSTAR
von Arthur Naiman – eine klar gegliederte Einführung, die aufzeigt, wie das Textbearbeitungsprogramm WORDSTAR funktioniert, was man damit tun kann und wie es eingesetzt wird. 240 Seiten, 36 Abbildungen, Best.-Nr.: **3019** (1983)

ERFOLG MIT VisiCalc
von D. Hergert – umfassende Einführung in VisiCalc und seine Anwendung. Zeigt Ihnen u. a.: Aufstellung eines Verteilungsbogens, Benutzung von VisiCalc-Formeln, Verwendung der DIF-Datei-Funktion. 224 Seiten, 58 Abbildungen, Best.-Nr.: **3030** (1983)

ERFOLG MIT MULTIPLAN
von Th. Ritter – das Tabellenkalkulations-Programm Multiplan hilft Ihnen bei der Lösung kommerzieller, wissenschaftlicher und allgemeiner Probleme. Lernen Sie die Möglichkeiten kennen, Ihre Software optimal zu nutzen! 208 Seiten, 68 Abbildungen, Best.-Nr.: **3043** (1984)

LOTUS 1-2-3. DATENVERARBEITUNG OHNE VORKENNTNISSE
von S. Heine – für alle, die ohne DV-Kenntnisse das starke Software-Paket LOTUS 1-2-3 für berufliche oder private Anwendungen einsetzen möchten. 264 Seiten, 64 Abb., Best.-Nr. **3052** (erscheint 1985)

ARBEITEN MIT LOTUS 1-2-3
von B. F. Kehlmann – die wichtigsten Anwendungsfunktionen von LOTUS 1-2-3 im Betrieb anhand praktischer Fallstudien. Ca. 200 Seiten, mit Abb., Best.-Nr. **3078** (erscheint 1985)

ARBEITEN MIT dBase II
von A. Simpson – Grundlagen und Programmiertechniken für die Datenbank-Verwaltung mit dBASE II. Zahlreiche praktische Tips. 264 Seiten, 50 Abb., Best.-Nr. **3070** (1984)



Fordern Sie ein Gesamtverzeichnis unserer Verlagsproduktion an:

SYBEX INC
2344 Sixth Street
Berkeley, CA 94710, USA
Tel.: (415) 848-8233
Telex: 336311

SYBEX-VERLAG GmbH
Vogelsanger Weg 111
4000 Düsseldorf 30
Tel.: (02 11) 6 18 02-0
Telex: 8 588 163

SYBEX
6–8, Impasse du Curé
75018 Paris
Tel.: 1/203-95-95
Telex: 211.801



CP/M Handbuch

Dieses Handbuch zu CP/M, dem meistgebrauchten Betriebssystem für Mikrocomputer, liegt bereits in der 6. Auflage vor und kann als das Standardwerk zu CP/M angesehen werden. Die Fachpresse empfahl es seiner Fülle und Klarheit wegen als wertvollen Begleiter bei der Arbeit mit dem Computer.

Aus dem Inhalt

- Einführung in CP/M und MP/M
- CP/M- und MP/M-Funktionen
- Handhabung von Dateien
- Anwendung des Editors
- Betriebssysteme der CP/M-Familie
- Kurzbeschreibungen der Befehle und Programme

Mit vielen praktischen Hinweisen ermöglicht das Buch dem Anfänger die Anwendung von CP/M mit allen Möglichkeiten. Fortgeschrittene erhalten ein umfassendes Nachschlagewerk über die Versionen CP/M 2.2, CP/M Plus, CP/M-86, Concurrent CP/M-86, MP/M II und MP/M-86.

Über den Autor

Dr. Rodnay Zaks ist ein Pionier in der Vermittlung von Computerwissen. Er war im „Silicon Valley" ein Wegbereiter für die industrielle Anwendung von Mikroprozessoren und gab jahrelang Unterricht über deren Entwicklung, Programmierung und Anwendung. Diese Erfahrung schlägt sich in seinen Publikationen nieder: Viele Bücher von Dr. Zaks wurden zu Standardsellern, die in mehr als 10 Sprachen vorliegen.

ISBN 3-88745-053-1